IEEE 802.1Q YANG Module Specifications

Document status: Draft
Issue number: 0.4
Issue date: May 12, 2015
Author: Marc Holness

Publication History

Issue	Section(s)	Status	Author	Date	Comments
0.1	All	Draft	Marc Holness	May 01, 2015	Initial draft
0.2		Draft	Marc Holness		Update YANG Structure
0.3		Drat	Marc Holness		Introduce UML model
0.4		Draft	Marc Holness	May 12, 2015	Include IETF Interface YANG module augmentation

Contents

List of figures

1 Introduction

This document defines a YANG [RFC6020] model for the configuration management of (a subset of) IEEE 802.1Q -2011 VLAN-aware Bridges. Specifically, base VLAN Bridges and Provider Bridges are scoped. The YANG module definitions, proposed here, are based on the existing IEEE 802.1Q-2011 bridge management model (specified in clause 12, which defines the set of managed objects, and their functionality, that allow administrative configuration of VLANs).

1.1 Scope, Model, and Applicability

The purpose of this document is to specify standards compliant IEEE 802.1Q VLAN and Provider Bridge implementations of YANG modules, for configuration management.

1.2 802.1Q and Netconf/YANG

Netconf is a widely accepted configuration management protocol that promises to simplify network configuration. YANG [RFC6020] is a formalized data modeling language used to model configuration and state data that can be used by Netconf [RFC6241]. The adoption of YANG, will allow IEEE 802.1 Bridging vendors as well as Network Management Systems to speak a common language, and thus simplify Service Provider operations.

Service Providers are moving towards a Netconf/YANG configuration management paradigm. As such vendors that provide IEEE 802.1Q Bridges and supporting functionality would be interested in this specification.

Currently, other SDOs (e.g., MEF and IETF) are developing YANG data models. IEEE 802 needs to provide specifications of YANG data models in support of IEE 802.1 managed entities. For example, MEF have defined YANG models for Service OAM Fault Management & Performance Monitoring, and IETF have defined YANG data models for Interface Management.

2 Modelling Assumptions and Constraints

1. “Port-and-protocol-based VLAN classification” feature is not supported.
2. “Extended Filtering Services” feature is not supported.
3. Bridge Protocol Entities not modelled.
4. MRP & MMRP are not modelled.
5. PBB & PBB-TE are not modelled.
6. TPMR not modelled.
7. CFM & DDCFM not modelled.
8. Time-sensitive streams are not modelled.
9. Congestion notification is not modelled.
10. SRP (Stream Reservation Protocol) is not modelled.
11.

3 802.1Q YANG Module Definitions

The YANG module definitions currently found in this document are based upon the Bridge Management configuration model specified in 802.1Q-2011).

Basic VLAN Bridge and Provide Bridge YANG module definitions are provided.

3.1 802.1Q Data Model Representation

Figure 2 illustrates the management information model for the 802.1Q components currently modelled.

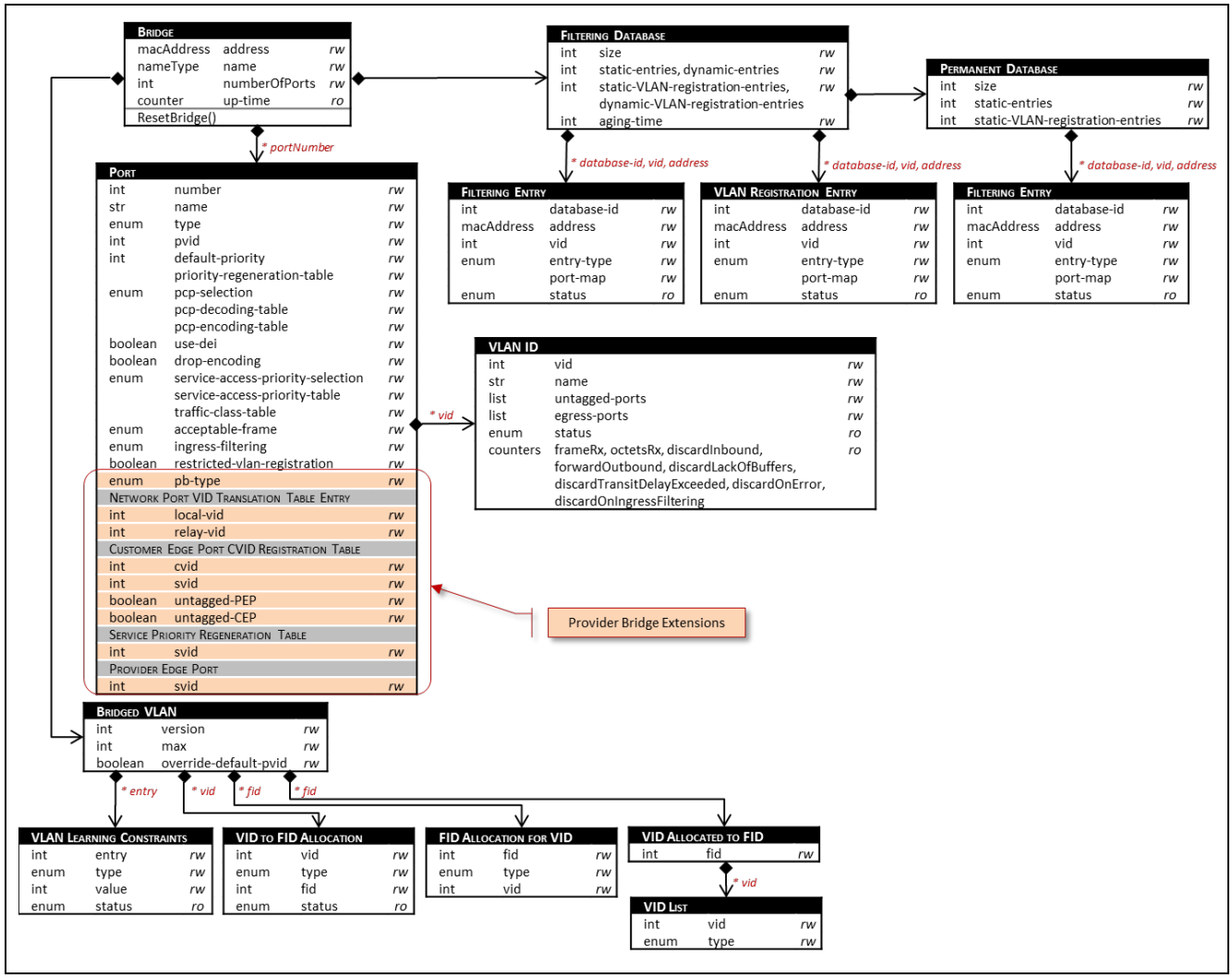

Figure 1: 802.1Q Bridge Management Information Model

3.2 YANG Data Model Schema

A simplified graphical representation of the data model is used to present the data scheme. 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.

3.3 YANG Modules NOT based on Augmentation of the IETF Interface YANG Module

3.3.1 Base VLAN Bridge YANG Tree

The YANG data model schema (or tree), in support of base VLAN Bridges, is graphically represented below.

```
module: dot1Q
   +--rw bridge
   |  +--rw bridge-information
   |  |  +--rw address?     ieee:mac-address
   |  |  +--rw name?        nameType
   |  |  +--rw num-ports?   portNumberType
   |  |  +--ro up-time?     yang:counter32
   |  +--rw port* [number]
   |  |  +--rw number                                portNumberType
   |  |  +--rw name?                                 nameType
   |  |  +--rw address?                              ieee:mac-address
   |  |  +--rw type?                                 enumeration
   |  |  +--rw pvid?                                 ieee:vlanid
   |  |  +--rw vlanId* [vid]
   |  |  |  +--rw vid               ieee:vlanid
   |  |  |  +--rw name?             nameType
   |  |  |  +--rw untagged-ports*   portNumberType
   |  |  |  +--rw egress-ports*     portNumberType
   |  |  |  +--ro status?           enumeration
   |  |  |  +--ro counters
   |  |  |     +--ro frame-rx?                          yang:counter32
   |  |  |     +--ro octets-rx?                         yang:counter64
   |  |  |     +--ro discard-inbound?                   yang:counter32
   |  |  |     +--ro forward-outbound?                  yang:counter32
   |  |  |     +--ro discard-lack-of-buffers?           yang:counter32
   |  |  |     +--ro discard-transit-delay-exceeded?    yang:counter32
   |  |  |     +--ro discard-on-error?                  yang:counter32
   |  |  |     +--ro discard-on-ingress-filtering?      yang:counter32 {IngressFiltering}?
   |  |  +--rw default-priority?                        priorityType
   |  |  +--rw priority-regeneration-table
   |  |  |  +--rw priority0?   priorityType
   |  |  |  +--rw priority1?   priorityType
   |  |  |  +--rw priority2?   priorityType
   |  |  |  +--rw priority3?   priorityType
   |  |  |  +--rw priority4?   priorityType
   |  |  |  +--rw priority5?   priorityType
   |  |  |  +--rw priority6?   priorityType
   |  |  |  +--rw priority7?   priorityType
   |  |  +--rw pcp-selection?                           pcpSelectionType
   |  |  +--rw pcp-decoding-table
   |  |  |  +--rw type?    pcpSelectionType
   |  |  |  +--rw table
   |  |  |     +--rw pcp0
   |  |  |     |  +--rw priority?        priorityType
   |  |  |     |  +--rw drop-eligible?   boolean
   |  |  |     +--rw pcp1
   |  |  |     |  +--rw priority?        priorityType
   |  |  |     |  +--rw drop-eligible?   boolean
   |  |  |     +--rw pcp2
   |  |  |     |  +--rw priority?        priorityType
   |  |  |     |  +--rw drop-eligible?   boolean
```

```
|  |  |     +--rw pcp3
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp4
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp5
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp6
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp7
|  |  |        +--rw priority?        priorityType
|  |  |        +--rw drop-eligible?   boolean
|  |  +--rw pcp-encoding-table
|  |  |  +--rw type?             pcpSelectionType
|  |  |  +--rw encoding-table
|  |  |     +--rw priority0
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority1
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority2
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority3
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority4
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority5
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority6
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority7
|  |  |        +--rw pcp-dei-false?   priorityType
|  |  |        +--rw pcp-dei-true?    priorityType
|  |  +--rw use-dei?                              boolean
|  |  +--rw drop-encoding?                        boolean
|  |  +--rw service-access-priority-selection?    serviceAccessPrioritySelectionType
|  |  +--rw service-access-priority-table
|  |  |  +--rw priority0?   priorityType
|  |  |  +--rw priority1?   priorityType
|  |  |  +--rw priority2?   priorityType
|  |  |  +--rw priority3?   priorityType
|  |  |  +--rw priority4?   priorityType
|  |  |  +--rw priority5?   priorityType
|  |  |  +--rw priority6?   priorityType
|  |  |  +--rw priority7?   priorityType
|  |  +--rw traffic-class-table
|  |  +--rw acceptable-frame?                     acceptableFrameType
|  |  +--rw ingress-filtering?                    ingressFilteringType
|  |  +--rw restricted-vlan-registration?         boolean
|  +--rw bridged-vlan
|     +--rw vlan
|     |  +--rw version?                 uint8
|     |  +--rw max?                     ieee:vlanid
|     |  +--rw override-default-pvid?   boolean
|     +--rw vlan-learning-constraints* [entry]
|     |  +--rw entry     uint16
|     |  +--rw type?     learningConstraintType
|     |  +--rw value?    uint16
|     |  +--ro status?   enumeration
|     +--rw vid-to-fid-allocation* [vid]
|     |  +--rw vid       ieee:vlanid
```

```
|     |  +--rw type?     VID-FID-allocationType
|     |  +--rw fid?      ieee:vlanid
|     |  +--ro status?   enumeration
|     +--rw fid-allocation-for-vid* [fid]
|     |  +--rw fid                  ieee:vlanid
|     |  +--rw vid?                 ieee:vlanid
|     |  +--rw vid-fid-allocation?  VID-FID-allocationType
|     +--rw vid-allocated-to-fid* [fid]
|        +--rw fid         ieee:vlanid
|        +--rw vid-list* [vid]
|           +--rw vid     ieee:vlanid
|           +--rw type?   VID-FID-allocationType
+--rw filtering-database
   +--rw information
   |  +--rw size?                              uint16
   |  +--rw static-entries?                    uint16
   |  +--rw dynamic-entries?                   uint16
   |  +--rw static-vlan-registration-entries?  uint16
   |  +--rw dynamic-vlan-registration-entries? uint16
   |  +--rw aging-time?                        yang:timeticks
   +--rw filtering-entry* [database-id vid address]
   |  +--rw database-id    uint16
   |  +--rw address        ieee:mac-address
   |  +--rw vid            ieee:vlanid
   |  +--rw entry-type?    enumeration
   |  +--ro status?        enumeration
   |  +--rw port-map* [port-number]
   |     +--rw port-number    portNumberType
   |     +--rw map?           enumeration
   +--rw vlan-registration-entry* [database-id vid address]
   |  +--rw database-id    uint16
   |  +--rw address        ieee:mac-address
   |  +--rw vid            ieee:vlanid
   |  +--rw entry-type?    enumeration
   |  +--ro status?        enumeration
   |  +--rw port-map* [port-number]
   |     +--rw port-number    portNumberType
   |     +--rw map?           enumeration
   +--rw permanent-database
      +--rw information
      |  +--rw size?                              uint16
      |  +--rw static-entries?                    uint16
      |  +--rw static-vlan-registration-entries?  uint16
      +--rw filtering-entry* [database-id vid address]
         +--rw database-id    uint16
         +--rw address        ieee:mac-address
         +--rw vid            ieee:vlanid
         +--rw entry-type?    enumeration
         +--ro status?        enumeration
         +--rw port-map* [port-number]
            +--rw port-number    portNumberType
            +--rw map?           enumeration
```

3.3.2 Provide Bridge YANG Tree

The YANG data model schema (or tree), in support of Provider Bridges, is graphically represented below.

```
module: dot1Q-PB
augment /dot1Q:bridge/dot1Q:port:
   +--rw pb-type?                                    dot1Q:portType
   +--rw network-port-vid-translation-table-entry
   |  +--rw local-vid?   ieee:vlanid
   |  +--rw relay-vid?   ieee:vlanid
   +--rw customer-edge-port-cvid-registration-table
   |  +--rw cvid?          ieee:vlanid
   |  +--rw svid?          ieee:vlanid
```

```
      |  +--rw untagged-PEP?   boolean
      |  +--rw untagged-CEP?   boolean
      +--rw provider-edge-port
      |  +--rw svid?   ieee:vlanid
      +--rw service-priority-regeneration-table
         +--rw svid?   ieee:vlanid
```

3.3.3 YANG Data Module Definition

The (draft) YANG module definition is partitioned into a two modules.

- Module (dot1Q.yang) contains the YANG module definitions in support of the base VLAN Bridge.
  - Sub-module dot1Q-types.yang contains the specific type definitions, related to VLAN Bridges.
- Module (dot1Q-PB.yang) contains the YANG module definitions in support of the Provider Bridges.

3.3.3.1 Module (dot1Q.yang) Definition

The dot1Q.yang main module is provided below:

```
module dot1Q {

    namespace "ieee:ns:yang:ieee-dot1Q-bridge";
    prefix "dot1Q";

    import ieee-types { prefix ieee; }
        import yang-types { prefix yang; }

        include dot1Q-types;

    organization
      "Institute of Electrical and Electronics Engineers";

    contact
      "Web URL: http://ieee.org/
          E-mail:  corporate-communications@ieee.org
          Postal:
                   U.S.A.
          Phone:   +1 732-563-6820
          Fax:     +1 732-981-9511";

        description
        "TBD.";

    revision "2015-03-10" {
      description
        "Initial Version.";
      reference
        "IEEE 802.1Q-2011, Media Access Control (MAC) Bridges and Virtual
         Bridged Local Area Networks.";
    }

        feature IngressFiltering {
     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 frame's 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.";
        }
        feature ExtendedFilteringServices {
                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 Port's default forwarding or filtering behavior
                with regard to group MAC addresses for which no specific static or
                dynamic filtering information has been configured.";
        }
        feature PortAndProtocolBasedVLANClassification {
                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.";
        }

        container bridge {
                description "The Bridge Configuration object models the operations
                        that modify, or inquire about, the configuration of the
                        Bridge's resources. There is a single Bridge Configuration
                        object per Bridge.";
                container bridge-information {
                        uses bridgeInfo;
                }
                list port {
                        key "number";
                        leaf number {
                                type portNumberType;
                        }
                        leaf name {
                                type nameType;
                        }
                        leaf address {
                                type ieee:mac-address;
                        }
                        leaf type {
                                type enumeration {
                                        enum IEEE-802.3;
                                        enum ISO-IEC-8802-4;
                                        enum ISO-IEC-8802-5;
                                        enum ISO-IEC-8802-6;
                                        enum ISO-IEC-8802-9;
                                        enum IEEE-802-9a;
                                        enum ISO-IEC-8802-11;
                                        enum ISO-IEC-8802-12-802.3;
                                        enum ISO-IEC-8802-12-802.5;
                                        enum ISO-IEC-9314;
                                        enum other;
                                }
                        }
                        leaf pvid {
                                type ieee:vlanid;
                        }
                        list vlanId {
                                key "vid";
                                leaf vid {
                                        type ieee:vlanid;
                                }
                                leaf name {
                                        type nameType;
                                }
                                leaf-list untagged-ports {
                                        type portNumberType;
                                }
                                leaf-list egress-ports {
```

```
                type portNumberType;
            }
            /* -- Not supported
            leaf protocol-based-vlan-classification {
                if-feature PortAndProtocolBasedVLANClassification;
                type boolean;
            }
            leaf max-vid-set-entries {
                if-feature PortAndProtocolBasedVLANClassification;
                type uint16;
            }
            container protocol-group {
                if-feature PortAndProtocolBasedVLANClassification;
                leaf protocol-group-id {
                    type uint16;
                }
                leaf vid {
                    type ieee:vlanid;
                }
            }
            */
            leaf status {
                config false;
                type enumeration {
                    enum reject-no-spare-vid;
                    enum reject-vid-out-of-range;
                    enum accepted;
                }
            }
            container counters {
                description "The Port Counters object models the operations that
                    can be performed on the Port counters of the Forwarding Process
                    resource. There are multiple instances (one for each VID for
                    each MAC Entity) of the Port Counters object per Bridge.";
                config false;
                leaf frame-rx {
                    type yang:counter32;
                }
                leaf octets-rx {
                    type yang:counter64;
                }
                leaf discard-inbound {
                    type yang:counter32;
                }
                leaf forward-outbound {
                    type yang:counter32;
                }
                leaf discard-lack-of-buffers {
                    type yang:counter32;
                }
                leaf discard-transit-delay-exceeded {
                    type yang:counter32;
                }
                leaf discard-on-error {
                    type yang:counter32;
                }
                leaf discard-on-ingress-filtering {
                    if-feature IngressFiltering;
                    type yang:counter32;
                }
            }
        }
        leaf default-priority {
            type priorityType;
        }
        container priority-regeneration-table {
            uses priorityRegenerationTable;
        }
        leaf pcp-selection {
            type pcpSelectionType;
```

```
        }
        container pcp-decoding-table {
                leaf type {
                        type pcpSelectionType;
                }
                container table {
                        uses pcpDecodingTable;
                }
        }
        container pcp-encoding-table {
                leaf type {
                        type pcpSelectionType;
                }
                container encoding-table {
                        uses pcpEncodingTable;
                }
        }
        leaf use-dei {
                type boolean;
        }
        leaf drop-encoding {
                type boolean;
        }
        leaf service-access-priority-selection {
                type serviceAccessPrioritySelectionType;
        }
        container service-access-priority-table {
                uses serviceAccessPriorityTable;
        }
        container traffic-class-table {
                uses trafficClassTable;
        }
        leaf acceptable-frame {
                description "To configure the Acceptable Frame Types parameter
                        associated with one or more Ports";
                type acceptableFrameType;
        }
        leaf ingress-filtering {
                description "To configure the Enabled Ingress Filtering parameter(s)
                        associated with one or more Ports";
                type ingressFilteringType;
        }
        leaf restricted-vlan-registration {
                description "To configure the Restricted_VLAN_Registration
                        parameter associated with one or more Ports.";
                type boolean;
        }
}
container bridged-vlan {
        description "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";
        container vlan {
                leaf version {
                        type uint8 {
                                range "1..2";
                        }
                }
                leaf max {
                        type ieee:vlanid;
                }
                leaf override-default-pvid {
                        type boolean;
                }
                /*
                leaf protocol-template-format {
                        if-feature PortAndProtocolBasedVLANClassification;
                        type protocolTemplateFormatType;
                }
```

```
        leaf max-protocol-group-database-entries {
                // Need to work on this section
                if-feature PortAndProtocolBasedVLANClassification;
                type uint16;
        }
        */
}
list vlan-learning-constraints {
        description "The VLAN Learning Constraints managed object models
                operations that modify, or inquire about, the set of VLAN
                Learning Constraints (8.8.8.2) and VID to FID allocations
                (8.8.8.1) that apply to the operation of the Learning Process
                and the Filtering Database. There is a single VLAN Learning
                Constraints managed object per Bridge. The object is modeled
                as a pair of fixed-length tables, as follows:
                        a) A Learning Constraint table in which each table
                           entry either defines a single Learning Constraint or
                           is undefined. For some of the operations that can be
                           performed on the table, an entry index is used; this
                           identifies the number of the entry in the table, where
                           index number 1 is the first, and N is the last (where
                           the table contains N entries).";
        key "entry";
        leaf entry {
                type uint16;
        }
        leaf type {
                type learningConstraintType;
        }
        leaf value {
                // Need to work on this one for sure! Needs to be a
                // container which depends on the learning constraints type.
                type uint16;
        }
        leaf status {
                config false;
                type enumeration {
                        enum rejected;
                        enum accepted;
                }
        }
}
list vid-to-fid-allocation {
        key "vid";
        leaf vid {
                type ieee:vlanid;
        }
        leaf type {
                type VID-FID-allocationType;
        }
        leaf fid {
                type ieee:vlanid;
        }
        leaf status {
                config false;
                type enumeration {
                        enum rejected-inconsistent-learning-constraint;
                        enum rejected-vid-exceed-max;
                        enum rejected-fid-exceed-max;
                        enum accepted;
                }
        }
}
list fid-allocation-for-vid {
        key "fid";
        leaf fid {
                type ieee:vlanid;
        }
        leaf vid {
                type ieee:vlanid;
```

```
                        }
                        leaf vid-fid-allocation {
                                type VID-FID-allocationType;
                        }
                }
                list vid-allocated-to-fid {
                        key "fid";
                        leaf fid {
                                type ieee:vlanid;
                        }
                        list vid-list {
                                key "vid";
                                leaf vid {
                                        type ieee:vlanid;
                                }
                                leaf type {
                                        type VID-FID-allocationType;
                                }
                        }
                }
        }
        /*
        container protocol-group-database {
                description "To configure a Protocol Group Database entry.";
                if-feature PortAndProtocolBasedVLANClassification;
                leaf format {
                        type protocolTemplateFormatType;
                }
                leaf index {
                        type union {
                                type etherType;
                                type protocolIdType;
                                type SAPAddressType;
                        }
                }
                leaf protocol-group-id {
                        type uint16;
                }
                leaf status {
                        type enumeration {
                                enum rejected-no-spare-protocol-group-database-entries;
                                enum rejected-unsupported-frame-format;
                                enum rejected-unsupported-ethertype-pid-dssap-ssap;
                                enum accepted;
                        }
                }
        }
        */
}

container filtering-database {
        description "The Filtering Database object models the operations that can
                be performed on, or affect, the Filtering Database as a whole. There is
                a single Filtering Database object per Bridge";
        container information {
                leaf size {
                        type uint16;
                }
                leaf static-entries {
                        type uint16;
                }
                leaf dynamic-entries {
                        type uint16;
                }
                leaf static-vlan-registration-entries {
                        type uint16;
                }
                leaf dynamic-vlan-registration-entries {
                        type uint16;
                }
```

```
        leaf aging-time {
                type yang:timeticks;
                units seconds;
                default 300;
        }
        /*
        leaf mac-address-registration-entries {
                if-feature ExtendedFilteringServices;
                type uint16;
        }
        */
}
list filtering-entry {
        description "A Static Filtering Entry object models the operations that
                can be performed on a single Static Filtering Entry in the Filtering
                Database. The set of Static Filtering Entry objects within the
                Filtering Database changes only under management control.

                A Dynamic Filtering Entry object models the operations that can be
                performed on a single Dynamic Filtering Entry (i.e., one that is
                created by the Learning Process as a result of the observation of
                network traffic) in the Filtering Database";
        key "database-id vid address";
        uses databaseIndex;
        leaf entry-type {
                type enumeration {
                        enum static;
                        enum dynamic;
                }
        }
        leaf status {
                config false;
                type enumeration {
                        enum accepted;
                        enum rejected;
                }
        }
        uses portMap;
}
list vlan-registration-entry {
        description "A VLAN Registration Entry object models the operations that
                can be performed on a single VLAN Registration Entry in the Filtering
                Database. The set of VLAN Registration Entry objects within the
                Filtering Database changes under management control and also as a
                result of MVRP exchanges";
        key "database-id vid address";
        uses databaseIndex;
        leaf entry-type {
                type enumeration {
                        enum static;
                        enum dynamic;
                }
        }
        leaf status {
                config false;
                type enumeration {
                        enum accepted;
                        enum rejected;
                }
        }
        uses portMap;
}
container permanent-database {
        description "The Permanent Database object models the operations
                that can be performed on, or affect, the Permanent Database.
                There is a single Permanent Database per Filtering Database.";
        container information {
                leaf size {
                        type uint16;
                }
```

```
                        leaf static-entries {
                                type uint16;
                        }
                        leaf static-vlan-registration-entries {
                                type uint16;
                        }
                }
                list filtering-entry {
                        key "database-id vid address";
                        uses databaseIndex;
                        leaf entry-type {
                                type enumeration {
                                        enum static;
                                        enum dynamic;
                                }
                        }
                        leaf status {
                                config false;
                                type enumeration {
                                        enum accepted;
                                        enum rejected;
                                }
                        }
                        uses portMap;
                }
        }
    }
}
```

3.3.3.2 Sub-module (dot1Q-type.yang) Definitions

The dot1Q-types.yang sub-module is provided below:

```
submodule dot1Q-types {

    belongs-to dot1Q { prefix dot1Q; }

    import ieee-types { prefix ieee; }
    import yang-types { prefix yang; }

    description "Common types used within dot1Q.";

    revision "2015-03-16" {
        description "Initial revision.";
    }

        typedef nameType {
                type string {
                        length "0..32";
                }
        }

        typedef portNumberType {
                type uint16;
        }

        typedef priorityType {
                type uint8 {
                        range "0..7";
                }
        }

        typedef pcpSelectionType {
                type enumeration {
                        enum 8P0D;
                        enum 7P1D;
                        enum 6P2D;
                        enum 5P3D;
```

```
        }
}

typedef acceptableFrameType {
        type enumeration {
                enum admit-only-VLAN-tagged-frames;
                enum admit-only-untagged-and-priority-tagged-frames;
                enum admit-all-frames;
        }
}

typedef ingressFilteringType {
        type enumeration {
                enum enabled;
                enum disabled;
        }
}
typedef serviceAccessPrioritySelectionType {
        type enumeration {
                enum enabled;
                enum disabled;
        }
}

typedef protocolTemplateFormatType {
        type enumeration {
                enum Ethernet;
                enum RFC1042;
                enum SNAP;
                enum LLC;
        }
}

typedef etherType {
        // To be defined.
        type string {
                pattern '[0-9a-fA-F]{2}-[0-9a-fA-F]{2}';
        }
}

typedef protocolIdType {
        // To be defined.
        type string {
                pattern '[0-9a-fA-F]{2}(-[0-9a-fA-F]{2}){4}';
        }
}

typedef SAPAddressType {
        type ieee:mac-address;
}

typedef learningConstraintType {
        type enumeration {
                enum undefined;
                enum S;
                enum I;
        }
}

typedef VID-FID-allocationType {
        type enumeration {
                enum undefined;
                enum fixed;
                enum dynamic;
        }
}
typedef portType {
        type enumeration {
                enum provider-nework;
                enum customer-network;
```

```
                enum customer-edge;
        }
}

grouping bridgeInfo {
        leaf address {
                type ieee:mac-address;
        }
        leaf name {
                type nameType;
        }
        leaf num-ports {
                type portNumberType;
        }
        leaf up-time {
                config false;
                type yang:counter32;
                units seconds;
        }
}

grouping priorityRegenerationTable {
        leaf priority0 {
                type priorityType;
        }
        leaf priority1 {
                type priorityType;
        }
        leaf priority2 {
                type priorityType;
        }
        leaf priority3 {
                type priorityType;
        }
        leaf priority4 {
                type priorityType;
        }
        leaf priority5 {
                type priorityType;
        }
        leaf priority6 {
                type priorityType;
        }
        leaf priority7 {
                type priorityType;
        }
}

grouping pcpDecodingTable {
        container pcp0 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp1 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp2 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
```

```
                        type boolean;
                }
        }
        container pcp3 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp4 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp5 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp6 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp7 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
}

grouping pcpEncodingTable {
        container priority0 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority1 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority2 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
```

```
        container priority3 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority4 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority5 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority6 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority7 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
}

grouping serviceAccessPriorityTable {
        leaf priority0 {
                type priorityType;
        }
        leaf priority1 {
                type priorityType;
        }
        leaf priority2 {
                type priorityType;
        }
        leaf priority3 {
                type priorityType;
        }
        leaf priority4 {
                type priorityType;
        }
        leaf priority5 {
                type priorityType;
        }
        leaf priority6 {
                type priorityType;
        }
        leaf priority7 {
                type priorityType;
        }
}

grouping trafficClassTable {
```

```
                // To be defined
        }

        grouping databaseIndex {
                leaf database-id {
                        type uint16;
                }
                leaf address {
                        type ieee:mac-address;
                }
                leaf vid {
                        type ieee:vlanid;
                }
        }

        grouping portMap {
                list port-map {
                        key "port-number";
                        leaf port-number {
                                type portNumberType;
                        }
                        leaf map {
                                // Need to work on this
                                type enumeration {
                                        enum forward;
                                        enum filter;
                                        enum forward-filter;
                                }
                        }
                }
        }
    }
```

3.3.3.3 Module (dot1Q-PB.yang) Definitions

The dot1Q-PB.yang module is provided below:

```
module dot1Q-PB {

        namespace "ieee:ns:yang:ieee-dot1Q-PB";
    prefix "dot1Q-PB";

        import dot1Q {prefix dot1Q; }
    import ieee-types { prefix ieee; }

    augment "/dot1Q:bridge/dot1Q:port" {
                leaf pb-type {
                        description "To obtain information regarding the designated type of an
                                externally accessible port on a Provider Bridge.";
                        type dot1Q:portType;
                }
                container network-port-vid-translation-table-entry {
                        description "The VID Translation Table, which provides a bidirectional
                                mapping between a local S-VID (used in data and protocol frames
                                transmitted and received through this Customer Network Port or
                                Provider Network Port) and a relay S-VID (used by the filtering
                                and forwarding processes of the S-VLAN component in a Provider
                                Bridge).";
                        leaf local-vid {
                                type ieee:vlanid;
                        }
                        leaf relay-vid {
                                type ieee:vlanid;
                        }
                }
                container customer-edge-port-cvid-registration-table {
                        description "The C-VID Registration Table, which provides a mapping
                                between a Customer VLAN Identifier (C-VID) and the service instance
```

```
                        represented by a Service VLAN Identifier (S-VID) selected for that
                        C-VLAN. This table provides the equivalent functionality of
                                1) Configuring the PVID of the internal Customer Network Port
                                   on the S-VLAN component;
                                2) Adding the corresponding Provider Edge Port on the C-VLAN
                                   component to the member set of the C-VLAN;
                                3) Adding the Provider Edge Port and/or Customer Edge Port to
                                   the untagged set of the C-VLAN (if it is desired that frames
                                   forwarded to that port are transmitted untagged for this
                                   C-VLAN).";
                leaf cvid {
                        type ieee:vlanid;
                }
                leaf svid {
                        type ieee:vlanid;
                }
                leaf untagged-PEP {
                        type boolean;
                }
                leaf untagged-CEP {
                        type boolean;
                }
        }
        container provider-edge-port {
                description "The Provider Edge Port configuration parameters,
                        which provide the subset of the Bridge VLAN
                        Configuration managed object (12.10.1) that is relevant
                        for the internal ports of the C-VLAN component
                        associated with the Customer Edge Port.";
                leaf svid {
                        type ieee:vlanid;
                }
        }
        container service-priority-regeneration-table {
                description "The Service Priority Regeneration Table, which
                        provides the Priority Regeneration Table (12.6.2) for each
                        internal Customer Network Port connected to the C-VLAN
                        component associated with the Customer Edge Port.";
                leaf svid {
                        type ieee:vlanid;
                }
        }
    }
}
```

3.4 YANG Modules Based on Augmentation of the IETF Interface YANG Module

3.4.1 Base VLAN Bridge YANG Tree

The YANG data model schema (or tree), in support of base VLAN Bridges, is graphically represented below.

```
module: dot1Qv2
   +--rw bridge
   |  +--rw bridge-information
   |  |  +--rw address?     ieee:mac-address
   |  |  +--rw name?        nameType
   |  |  +--rw num-ports?   portNumberType
   |  |  +--ro up-time?     yang:counter32
   |  +--rw interfaces* [name]
   |  |  +--rw name                                -> /if:interfaces/interface/name
   |  |  +--rw address?                            ieee:mac-address
   |  |  +--rw type?                               enumeration
   |  |  +--rw pvid?                               ieee:vlanid
   |  |  +--rw vlanId* [vid]
   |  |  |  +--rw vid               ieee:vlanid
   |  |  |  +--rw name?             nameType
```

```
|  |  |  +--rw untagged-ports*   portNumberType
|  |  |  +--rw egress-ports*     portNumberType
|  |  +--rw default-priority?                    priorityType
|  |  +--rw priority-regeneration-table
|  |  |  +--rw priority0?   priorityType
|  |  |  +--rw priority1?   priorityType
|  |  |  +--rw priority2?   priorityType
|  |  |  +--rw priority3?   priorityType
|  |  |  +--rw priority4?   priorityType
|  |  |  +--rw priority5?   priorityType
|  |  |  +--rw priority6?   priorityType
|  |  |  +--rw priority7?   priorityType
|  |  +--rw pcp-selection?                       pcpSelectionType
|  |  +--rw pcp-decoding-table
|  |  |  +--rw type?    pcpSelectionType
|  |  |  +--rw table
|  |  |     +--rw pcp0
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp1
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp2
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp3
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp4
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp5
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp6
|  |  |     |  +--rw priority?        priorityType
|  |  |     |  +--rw drop-eligible?   boolean
|  |  |     +--rw pcp7
|  |  |        +--rw priority?        priorityType
|  |  |        +--rw drop-eligible?   boolean
|  |  +--rw pcp-encoding-table
|  |  |  +--rw type?             pcpSelectionType
|  |  |  +--rw encoding-table
|  |  |     +--rw priority0
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority1
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority2
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority3
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority4
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority5
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority6
|  |  |     |  +--rw pcp-dei-false?   priorityType
|  |  |     |  +--rw pcp-dei-true?    priorityType
|  |  |     +--rw priority7
|  |  |        +--rw pcp-dei-false?   priorityType
|  |  |        +--rw pcp-dei-true?    priorityType
|  |  +--rw use-dei?                             boolean
|  |  +--rw drop-encoding?                       boolean
|  |  +--rw service-access-priority-selection?   serviceAccessPrioritySelectionType
```

```
|  |  +--rw service-access-priority-table
|  |  |  +--rw priority0?   priorityType
|  |  |  +--rw priority1?   priorityType
|  |  |  +--rw priority2?   priorityType
|  |  |  +--rw priority3?   priorityType
|  |  |  +--rw priority4?   priorityType
|  |  |  +--rw priority5?   priorityType
|  |  |  +--rw priority6?   priorityType
|  |  |  +--rw priority7?   priorityType
|  |  +--rw traffic-class-table
|  |  +--rw acceptable-frame?                    acceptableFrameType
|  |  +--rw ingress-filtering?                   ingressFilteringType
|  |  +--rw restricted-vlan-registration?        boolean
|  +--ro interfaces-state* [name]
|  |  +--ro name      -> /if:interfaces-state/interface/name
|  |  +--ro vlanId* [vid]
|  |     +--ro vid         ieee:vlanid
|  |     +--ro status?     enumeration
|  |     +--ro counters
|  |        +--ro frame-rx?                         yang:counter32
|  |        +--ro octets-rx?                        yang:counter64
|  |        +--ro discard-inbound?                  yang:counter32
|  |        +--ro forward-outbound?                 yang:counter32
|  |        +--ro discard-lack-of-buffers?          yang:counter32
|  |        +--ro discard-transit-delay-exceeded?   yang:counter32
|  |        +--ro discard-on-error?                 yang:counter32
|  |        +--ro discard-on-ingress-filtering?     yang:counter32 {IngressFiltering}?
|  +--rw bridged-vlan
|     +--rw vlan
|     |  +--rw version?                uint8
|     |  +--rw max?                    ieee:vlanid
|     |  +--rw override-default-pvid?  boolean
|     +--rw vlan-learning-constraints* [entry]
|     |  +--rw entry     uint16
|     |  +--rw type?     learningConstraintType
|     |  +--rw value?    uint16
|     |  +--ro status?   enumeration
|     +--rw vid-to-fid-allocation* [vid]
|     |  +--rw vid       ieee:vlanid
|     |  +--rw type?     VID-FID-allocationType
|     |  +--rw fid?      ieee:vlanid
|     |  +--ro status?   enumeration
|     +--rw fid-allocation-for-vid* [fid]
|     |  +--rw fid                  ieee:vlanid
|     |  +--rw vid?                 ieee:vlanid
|     |  +--rw vid-fid-allocation?  VID-FID-allocationType
|     +--rw vid-allocated-to-fid* [fid]
|        +--rw fid         ieee:vlanid
|        +--rw vid-list* [vid]
|           +--rw vid     ieee:vlanid
|           +--rw type?   VID-FID-allocationType
+--rw filtering-database
   +--rw information
   |  +--rw size?                                uint16
   |  +--rw static-entries?                      uint16
   |  +--rw dynamic-entries?                     uint16
   |  +--rw static-vlan-registration-entries?    uint16
   |  +--rw dynamic-vlan-registration-entries?   uint16
   |  +--rw aging-time?                          yang:timeticks
   +--rw filtering-entry* [database-id vid address]
   |  +--rw database-id    uint16
   |  +--rw address        ieee:mac-address
   |  +--rw vid            ieee:vlanid
   |  +--rw entry-type?    enumeration
   |  +--ro status?        enumeration
   |  +--rw port-map* [port-number]
   |     +--rw port-number    portNumberType
   |     +--rw map?           enumeration
   +--rw vlan-registration-entry* [database-id vid address]
   |  +--rw database-id    uint16
```

```
|  +--rw address        ieee:mac-address
|  +--rw vid            ieee:vlanid
|  +--rw entry-type?    enumeration
|  +--ro status?        enumeration
|  +--rw port-map* [port-number]
|     +--rw port-number    portNumberType
|     +--rw map?           enumeration
+--rw permanent-database
   +--rw information
   |  +--rw size?                              uint16
   |  +--rw static-entries?                    uint16
   |  +--rw static-vlan-registration-entries?  uint16
   +--rw filtering-entry* [database-id vid address]
      +--rw database-id    uint16
      +--rw address        ieee:mac-address
      +--rw vid            ieee:vlanid
      +--rw entry-type?    enumeration
      +--ro status?        enumeration
      +--rw port-map* [port-number]
         +--rw port-number    portNumberType
         +--rw map?           enumeration
```

3.4.2 Provide Bridge YANG Tree

The YANG data model schema (or tree), in support of Provider Bridges, is graphically represented below.

```
module: dot1Qv2-PB
augment /dot1Q:bridge/dot1Q:interfaces:
   +--rw pb-type?                                    dot1Q:portType
   +--rw network-port-vid-translation-table-entry
   |  +--rw local-vid?   ieee:vlanid
   |  +--rw relay-vid?   ieee:vlanid
   +--rw customer-edge-port-cvid-registration-table
   |  +--rw cvid?           ieee:vlanid
   |  +--rw svid?           ieee:vlanid
   |  +--rw untagged-PEP?   boolean
   |  +--rw untagged-CEP?   boolean
   +--rw provider-edge-port
   |  +--rw svid?   ieee:vlanid
   +--rw service-priority-regeneration-table
      +--rw svid?   ieee:vlanid
```

3.4.3 YANG Data Module Definition

The (draft) YANG module definition is partitioned into a two modules.

- Module (dot1Qv2.yang) contains the YANG module definitions in support of the base VLAN Bridge.
  - Sub-module dot1Qv2-types.yang contains the specific type definitions, related to VLAN Bridges.
- Module (dot1Q-PBv2.yang) contains the YANG module definitions in support of the Provider Bridges.

3.4.3.1 Module (dot1Qv2.yang) Definition

The dot1Qv2.yang main module is provided below:

```
module dot1Qv2 {

    namespace "ieee:ns:yang:ieee-dot1Q-bridge";
    prefix "dot1Q";

    import ieee-types { prefix ieee; }
        import yang-types { prefix yang; }
```

```
    import ietf-interfaces { prefix "if"; }

    include dot1Qv2-types;

  organization
    "Institute of Electrical and Electronics Engineers";

  contact
    "Web URL: http://ieee.org/
        E-mail:  corporate-communications@ieee.org
        Postal:
                 U.S.A.
        Phone:   +1 732-563-6820
        Fax:     +1 732-981-9511";

    description
    "TBD.";

  revision "2015-03-10" {
    description
      "Initial Version.";
    reference
      "IEEE 802.1Q-2011, Media Access Control (MAC) Bridges and Virtual
       Bridged Local Area Networks.";
  }

    feature IngressFiltering {
   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 frame's 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.";
    }
    feature ExtendedFilteringServices {
            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 Port's default forwarding or filtering behavior
                    with regard to group MAC addresses for which no specific static or
                    dynamic filtering information has been configured.";
    }
    feature PortAndProtocolBasedVLANClassification {
            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.";
    }

    container bridge {
            description "The Bridge Configuration object models the operations
                    that modify, or inquire about, the configuration of the
                    Bridge's resources. There is a single Bridge Configuration
                    object per Bridge.";
            container bridge-information {
                    uses bridgeInfo;
            }
            list interfaces {
```

```
            key "name";
leaf name {
    type leafref {
        path "/if:interfaces/if:interface/if:name";
    }
}
            leaf address {
                    type ieee:mac-address;
            }
            leaf type {
                    type enumeration {
                            enum IEEE-802.3;
                            enum ISO-IEC-8802-4;
                            enum ISO-IEC-8802-5;
                            enum ISO-IEC-8802-6;
                            enum ISO-IEC-8802-9;
                            enum IEEE-802-9a;
                            enum ISO-IEC-8802-11;
                            enum ISO-IEC-8802-12-802.3;
                            enum ISO-IEC-8802-12-802.5;
                            enum ISO-IEC-9314;
                            enum other;
                    }
            }
            leaf pvid {
                    type ieee:vlanid;
            }
            list vlanId {
                    key "vid";
                    leaf vid {
                            type ieee:vlanid;
                    }
                    leaf name {
                            type nameType;
                    }
                    leaf-list untagged-ports {
                            type portNumberType;
                    }
                    leaf-list egress-ports {
                            type portNumberType;
                    }
                    /* -- Not supported
                    leaf protocol-based-vlan-classification {
                            if-feature PortAndProtocolBasedVLANClassification;
                            type boolean;
                    }
                    leaf max-vid-set-entries {
                            if-feature PortAndProtocolBasedVLANClassification;
                            type uint16;
                    }
                    container protocol-group {
                            if-feature PortAndProtocolBasedVLANClassification;
                            leaf protocol-group-id {
                                    type uint16;
                            }
                            leaf vid {
                                    type ieee:vlanid;
                            }
                    }
                    */
            }
            leaf default-priority {
                    type priorityType;
            }
            container priority-regeneration-table {
                    uses priorityRegenerationTable;
            }
            leaf pcp-selection {
                    type pcpSelectionType;
            }
```

```
            container pcp-decoding-table {
                    leaf type {
                            type pcpSelectionType;
                    }
                    container table {
                            uses pcpDecodingTable;
                    }
            }
            container pcp-encoding-table {
                    leaf type {
                            type pcpSelectionType;
                    }
                    container encoding-table {
                            uses pcpEncodingTable;
                    }
            }
            leaf use-dei {
                    type boolean;
            }
            leaf drop-encoding {
                    type boolean;
            }
            leaf service-access-priority-selection {
                    type serviceAccessPrioritySelectionType;
            }
            container service-access-priority-table {
                    uses serviceAccessPriorityTable;
            }
            container traffic-class-table {
                    uses trafficClassTable;
            }
            leaf acceptable-frame {
                    description "To configure the Acceptable Frame Types parameter
                            associated with one or more Ports";
                    type acceptableFrameType;
            }
            leaf ingress-filtering {
                    description "To configure the Enabled Ingress Filtering parameter(s)
                            associated with one or more Ports";
                    type ingressFilteringType;
            }
            leaf restricted-vlan-registration {
                    description "To configure the Restricted_VLAN_Registration
                            parameter associated with one or more Ports.";
                    type boolean;
            }
    }
    list interfaces-state {
            key "name";
leaf name {
    type leafref {
        path "/if:interfaces-state/if:interface/if:name";
    }
}
config false;
            list vlanId {
                    key "vid";
                    leaf vid {
                            type ieee:vlanid;
                    }
                    leaf status {
                            type enumeration {
                                    enum reject-no-spare-vid;
                                    enum reject-vid-out-of-range;
                                    enum accepted;
                            }
                    }
                    container counters {
                            description "The Port Counters object models the operations that
                                    can be performed on the Port counters of the Forwarding Process
```

```
                                resource. There are multiple instances (one for each VID for
                                each MAC Entity) of the Port Counters object per Bridge.";
                        leaf frame-rx {
                                type yang:counter32;
                        }
                        leaf octets-rx {
                                type yang:counter64;
                        }
                        leaf discard-inbound {
                                type yang:counter32;
                        }
                        leaf forward-outbound {
                                type yang:counter32;
                        }
                        leaf discard-lack-of-buffers {
                                type yang:counter32;
                        }
                        leaf discard-transit-delay-exceeded {
                                type yang:counter32;
                        }
                        leaf discard-on-error {
                                type yang:counter32;
                        }
                        leaf discard-on-ingress-filtering {
                                if-feature IngressFiltering;
                                type yang:counter32;
                        }
                }
        }
}
container bridged-vlan {
        description "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";
        container vlan {
                leaf version {
                        type uint8 {
                                range "1..2";
                        }
                }
                leaf max {
                        type ieee:vlanid;
                }
                leaf override-default-pvid {
                        type boolean;
                }
                /*
                leaf protocol-template-format {
                        if-feature PortAndProtocolBasedVLANClassification;
                        type protocolTemplateFormatType;
                }
                leaf max-protocol-group-database-entries {
                        // Need to work on this section
                        if-feature PortAndProtocolBasedVLANClassification;
                        type uint16;
                }
                */
        }
        list vlan-learning-constraints {
                description "The VLAN Learning Constraints managed object models
                        operations that modify, or inquire about, the set of VLAN
                        Learning Constraints (8.8.8.2) and VID to FID allocations
                        (8.8.8.1) that apply to the operation of the Learning Process
                        and the Filtering Database. There is a single VLAN Learning
                        Constraints managed object per Bridge. The object is modeled
                        as a pair of fixed-length tables, as follows:
                                a) A Learning Constraint table in which each table
                                   entry either defines a single Learning Constraint or
                                   is undefined. For some of the operations that can be
```

```
                                performed on the table, an entry index is used; this
                                identifies the number of the entry in the table, where
                                index number 1 is the first, and N is the last (where
                                the table contains N entries).";
                key "entry";
                leaf entry {
                        type uint16;
                }
                leaf type {
                        type learningConstraintType;
                }
                leaf value {
                        // Need to work on this one for sure! Needs to be a
                        // container which depends on the learning constraints type.
                        type uint16;
                }
                leaf status {
                        config false;
                        type enumeration {
                                enum rejected;
                                enum accepted;
                        }
                }
        }
        list vid-to-fid-allocation {
                key "vid";
                leaf vid {
                        type ieee:vlanid;
                }
                leaf type {
                        type VID-FID-allocationType;
                }
                leaf fid {
                        type ieee:vlanid;
                }
                leaf status {
                        config false;
                        type enumeration {
                                enum rejected-inconsistent-learning-constraint;
                                enum rejected-vid-exceed-max;
                                enum rejected-fid-exceed-max;
                                enum accepted;
                        }
                }
        }
        list fid-allocation-for-vid {
                key "fid";
                leaf fid {
                        type ieee:vlanid;
                }
                leaf vid {
                        type ieee:vlanid;
                }
                leaf vid-fid-allocation {
                        type VID-FID-allocationType;
                }
        }
        list vid-allocated-to-fid {
                key "fid";
                leaf fid {
                        type ieee:vlanid;
                }
                list vid-list {
                        key "vid";
                        leaf vid {
                                type ieee:vlanid;
                        }
                        leaf type {
                                type VID-FID-allocationType;
                        }
```

```
                        }
                }
        }
        /*
        container protocol-group-database {
                description "To configure a Protocol Group Database entry.";
                if-feature PortAndProtocolBasedVLANClassification;
                leaf format {
                        type protocolTemplateFormatType;
                }
                leaf index {
                        type union {
                                type etherType;
                                type protocolIdType;
                                type SAPAddressType;
                        }
                }
                leaf protocol-group-id {
                        type uint16;
                }
                leaf status {
                        type enumeration {
                                enum rejected-no-spare-protocol-group-database-entries;
                                enum rejected-unsupported-frame-format;
                                enum rejected-unsupported-ethertype-pid-dssap-ssap;
                                enum accepted;
                        }
                }
        }
        */
}

container filtering-database {
        description "The Filtering Database object models the operations that can
                be performed on, or affect, the Filtering Database as a whole. There is
                a single Filtering Database object per Bridge";
        container information {
                leaf size {
                        type uint16;
                }
                leaf static-entries {
                        type uint16;
                }
                leaf dynamic-entries {
                        type uint16;
                }
                leaf static-vlan-registration-entries {
                        type uint16;
                }
                leaf dynamic-vlan-registration-entries {
                        type uint16;
                }
                leaf aging-time {
                        type yang:timeticks;
                        units seconds;
                        default 300;
                }
                /*
                leaf mac-address-registration-entries {
                        if-feature ExtendedFilteringServices;
                        type uint16;
                }
                */
        }
        list filtering-entry {
                description "A Static Filtering Entry object models the operations that
                        can be performed on a single Static Filtering Entry in the Filtering
                        Database. The set of Static Filtering Entry objects within the
                        Filtering Database changes only under management control.
```

```
                A Dynamic Filtering Entry object models the operations that can be
                performed on a single Dynamic Filtering Entry (i.e., one that is
                created by the Learning Process as a result of the observation of
                network traffic) in the Filtering Database";
        key "database-id vid address";
        uses databaseIndex;
        leaf entry-type {
                type enumeration {
                        enum static;
                        enum dynamic;
                }
        }
        leaf status {
                config false;
                type enumeration {
                        enum accepted;
                        enum rejected;
                }
        }
        uses portMap;
}
list vlan-registration-entry {
        description "A VLAN Registration Entry object models the operations that
                can be performed on a single VLAN Registration Entry in the Filtering
                Database. The set of VLAN Registration Entry objects within the
                Filtering Database changes under management control and also as a
                result of MVRP exchanges";
        key "database-id vid address";
        uses databaseIndex;
        leaf entry-type {
                type enumeration {
                        enum static;
                        enum dynamic;
                }
        }
        leaf status {
                config false;
                type enumeration {
                        enum accepted;
                        enum rejected;
                }
        }
        uses portMap;
}
container permanent-database {
        description "The Permanent Database object models the operations
                that can be performed on, or affect, the Permanent Database.
                There is a single Permanent Database per Filtering Database.";
        container information {
                leaf size {
                        type uint16;
                }
                leaf static-entries {
                        type uint16;
                }
                leaf static-vlan-registration-entries {
                        type uint16;
                }
        }
        list filtering-entry {
                key "database-id vid address";
                uses databaseIndex;
                leaf entry-type {
                        type enumeration {
                                enum static;
                                enum dynamic;
                        }
                }
                leaf status {
                        config false;
```

```
                                        type enumeration {
                                                enum accepted;
                                                enum rejected;
                                        }
                                }
                                uses portMap;
                        }
                }
        }
    }
```

3.4.3.2 Sub-module (dot1Qv2-type.yang) Definitions

The dot1Qv2-types.yang sub-module is provided below:

```
submodule dot1Qv2-types {

    belongs-to dot1Qv2 { prefix dot1Q; }

    import ieee-types { prefix ieee; }
    import yang-types { prefix yang; }

    description "Common types used within dot1Q.";

    revision "2015-03-16" {
        description "Initial revision.";
    }

        typedef nameType {
                type string {
                        length "0..32";
                }
        }

        typedef portNumberType {
                type uint16;
        }

        typedef priorityType {
                type uint8 {
                        range "0..7";
                }
        }

        typedef pcpSelectionType {
                type enumeration {
                        enum 8P0D;
                        enum 7P1D;
                        enum 6P2D;
                        enum 5P3D;
                }
        }

        typedef acceptableFrameType {
                type enumeration {
                        enum admit-only-VLAN-tagged-frames;
                        enum admit-only-untagged-and-priority-tagged-frames;
                        enum admit-all-frames;
                }
        }

        typedef ingressFilteringType {
                type enumeration {
                        enum enabled;
                        enum disabled;
                }
        }
        typedef serviceAccessPrioritySelectionType {
```

```
        type enumeration {
                enum enabled;
                enum disabled;
        }
}

typedef protocolTemplateFormatType {
        type enumeration {
                enum Ethernet;
                enum RFC1042;
                enum SNAP;
                enum LLC;
        }
}

typedef etherType {
        // To be defined.
        type string {
                pattern '[0-9a-fA-F]{2}-[0-9a-fA-F]{2}';
        }
}

typedef protocolIdType {
        // To be defined.
        type string {
                pattern '[0-9a-fA-F]{2}(-[0-9a-fA-F]{2}){4}';
        }
}

typedef SAPAddressType {
        type ieee:mac-address;
}

typedef learningConstraintType {
        type enumeration {
                enum undefined;
                enum S;
                enum I;
        }
}

typedef VID-FID-allocationType {
        type enumeration {
                enum undefined;
                enum fixed;
                enum dynamic;
        }
}
typedef portType {
        type enumeration {
                enum provider-nework;
                enum customer-network;
                enum customer-edge;
        }
}

grouping bridgeInfo {
        leaf address {
                type ieee:mac-address;
        }
        leaf name {
                type nameType;
        }
        leaf num-ports {
                type portNumberType;
        }
        leaf up-time {
                config false;
                type yang:counter32;
                units seconds;
```

```
        }
}

grouping priorityRegenerationTable {
        leaf priority0 {
                type priorityType;
        }
        leaf priority1 {
                type priorityType;
        }
        leaf priority2 {
                type priorityType;
        }
        leaf priority3 {
                type priorityType;
        }
        leaf priority4 {
                type priorityType;
        }
        leaf priority5 {
                type priorityType;
        }
        leaf priority6 {
                type priorityType;
        }
        leaf priority7 {
                type priorityType;
        }
}

grouping pcpDecodingTable {
        container pcp0 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp1 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp2 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp3 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp4 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
```

```
        }
        container pcp5 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp6 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
        container pcp7 {
                leaf priority {
                        type priorityType;
                }
                leaf drop-eligible {
                        type boolean;
                }
        }
}

grouping pcpEncodingTable {
        container priority0 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority1 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority2 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority3 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority4 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority5 {
                leaf pcp-dei-false {
```

```
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority6 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
        container priority7 {
                leaf pcp-dei-false {
                        type priorityType;
                }
                leaf pcp-dei-true {
                        type priorityType;
                }
        }
}

grouping serviceAccessPriorityTable {
        leaf priority0 {
                type priorityType;
        }
        leaf priority1 {
                type priorityType;
        }
        leaf priority2 {
                type priorityType;
        }
        leaf priority3 {
                type priorityType;
        }
        leaf priority4 {
                type priorityType;
        }
        leaf priority5 {
                type priorityType;
        }
        leaf priority6 {
                type priorityType;
        }
        leaf priority7 {
                type priorityType;
        }
}

grouping trafficClassTable {
        // To be defined
}

grouping databaseIndex {
        leaf database-id {
                type uint16;
        }
        leaf address {
                type ieee:mac-address;
        }
        leaf vid {
                type ieee:vlanid;
        }
}

grouping portMap {
        list port-map {
                key "port-number";
```

```
                        leaf port-number {
                                type portNumberType;
                        }
                        leaf map {
                                // Need to work on this
                                type enumeration {
                                        enum forward;
                                        enum filter;
                                        enum forward-filter;
                                }
                        }
                }
        }
    }
```

3.4.3.3 Module (dot1Qv2-PB.yang) Definitions

The dot1Qv2-PB.yang module is provided below:

```
module dot1Qv2-PB {

        namespace "ieee:ns:yang:ieee-dot1Q-PB";
        prefix "dot1Q-PB";

        import dot1Qv2 {prefix dot1Q; }
                import ieee-types { prefix ieee; }

        augment "/dot1Q:bridge/dot1Q:interfaces" {
                leaf pb-type {
                        description "To obtain information regarding the designated type of an
                                externally accessible port on a Provider Bridge.";
                        type dot1Q:portType;
                }
                container network-port-vid-translation-table-entry {
                        description "The VID Translation Table, which provides a bidirectional
                                mapping between a local S-VID (used in data and protocol frames
                                transmitted and received through this Customer Network Port or
                                Provider Network Port) and a relay S-VID (used by the filtering
                                and forwarding processes of the S-VLAN component in a Provider
                                Bridge).";
                        leaf local-vid {
                                type ieee:vlanid;
                        }
                        leaf relay-vid {
                                type ieee:vlanid;
                        }
                }
                container customer-edge-port-cvid-registration-table {
                        description "The C-VID Registration Table, which provides a mapping
                                between a Customer VLAN Identifier (C-VID) and the service instance
                                represented by a Service VLAN Identifier (S-VID) selected for that
                                C-VLAN. This table provides the equivalent functionality of
                                        1) Configuring the PVID of the internal Customer Network Port
                                           on the S-VLAN component;
                                        2) Adding the corresponding Provider Edge Port on the C-VLAN
                                           component to the member set of the C-VLAN;
                                        3) Adding the Provider Edge Port and/or Customer Edge Port to
                                           the untagged set of the C-VLAN (if it is desired that frames
                                           forwarded to that port are transmitted untagged for this
                                           C-VLAN).";
                        leaf cvid {
                                type ieee:vlanid;
                        }
                        leaf svid {
                                type ieee:vlanid;
                        }
                        leaf untagged-PEP {
                                type boolean;
```

```
                }
                leaf untagged-CEP {
                        type boolean;
                }
        }
        container provider-edge-port {
                description "The Provider Edge Port configuration parameters,
                        which provide the subset of the Bridge VLAN
                        Configuration managed object (12.10.1) that is relevant
                        for the internal ports of the C-VLAN component
                        associated with the Customer Edge Port.";
                leaf svid {
                        type ieee:vlanid;
                }
        }
        container service-priority-regeneration-table {
                description "The Service Priority Regeneration Table, which
                        provides the Priority Regeneration Table (12.6.2) for each
                        internal Customer Network Port connected to the C-VLAN
                        component associated with the Customer Edge Port.";
                leaf svid {
                        type ieee:vlanid;
                }
        }
    }
}
```

4 Data Model Configuration Examples

This section presents examples of configuring 802.1Q VLAN Bridges.

Editor's note: Work in progress. Further work needed to rationalize.

4.1 Set Bridge Name

The following configuration example shows how to associate a test string with the Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns="http://.../config">
           <bridge>
              <name>"example-bridge-name"</name>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.2 Read Bridge

The following configuration example shows how to obtain general information regarding the Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <get>
     <source>
        <running/>
```

```
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <bridge-information/>
         </top>
      </filter>
   </get>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <data>
      <top xmlns=”http://.../config”>
         <bridge-information>
            <address>01:02:03:04:05:06</address>
            <name>”example-bridge-name”</name>
            <up-time>0</up-time>
         </bridge-information>
      </top>
   </data>
</rpc-reply>
```

4.3 Discover Bridge

The following configuration example shows how to solicit information regarding the Bridge(s) in the network.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <get>
      <source>
         <running/>
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <bridge>
               <bridge-information>
                    <address>01:02:03:04:05:06</address>
                    <name/>
                    <num-ports/>
                    <up-time/>
               </bridge-information>
            </bridge>
         </top>
      </filter>
   </get>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <data>
      <top xmlns=”http://.../config”>
         <bridge-information>
            <address>01:02:03:04:05:06</address>
            <name>”example-bridge-name”</name>
            <num-ports>24</num-ports>
            <up-time>0</up-time>
         </bridge-information>
      </top>
   </data>
</rpc-reply>
```

4.4 Set Bridge Port Name

The following configuration example shows how to obtain general information regarding a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <bridge>
              <port>
                  <number>1</number>
                  <name>”port-1”</name>
              </port>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.5 Read Bridge Port

The following configuration example shows how to obtain general information regarding a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <port>
              <number>1</number>
           <port/>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <port>
           <number>1</number>
           <name>”port-1”</name>
           <type>IEEE-802.3</type>
           <pvid>4000</pvid>
        </port>
     </top>
  </data>
</rpc-reply>
```

4.6 Read Port Counters

The following configuration example shows how to read the forwarding counters associated with a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <port>
              <number>1</number>
              <vlanID>
                   <vid>100</vid>
              </vlanID>
              </counters>
           <port/>
        </top>
     </filter>
  </get>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <port>
           <number>1</number>
           <name>”port-1”</name>
           <type>IEEE-802.3</type>
           <vlanID>
              <vid>100</vid>
              <counters>
                   <frameRx>1234</frameRx>
                   <octetRx>163456</octetRx>
                   <discardInbound>1</discardInbound>
                   <fowardOutbound>1230</fowardOutbound>
                   <discardLackOfBuffers>0</discardLackOfBuffers>
                   <discardTransitDelayExceeded>0</discardTransitDelayExceeded>
                   <discardOnError>0</discardOnError>
                   <discardOnIngressFiltering>0</discardOnIngressFiltering>
              </counters>
           </vlanID>
        </port>
     </top>
  </data>
</rpc-reply>
```

4.7 Read Port Default Priority

The following configuration example shows how to read the current state of the Default Priority parameter for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
```

```
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <port>
               <number>1</number>
               </default-priority>
            <port/>
         </top>
      </filter>
   </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <data>
      <top xmlns=”http://.../config”>
         <port>
            <number>1</number>
            <default-priority>0</default-priority>
         </port>
      </top>
   </data>
</rpc-reply>
```

4.8 Set Port Default Priority

The following configuration example shows how to set the current state of the Default Priority parameter for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <edit-config>
      <target>
         <running/>
      </target>
      <config>
         <top xmlns=”http://.../config”>
            <bridge>
               <port>
                    <number>1</number>
                    <default-priority>0</default-priority>
               </port>
            </bridge>
         </top>
      </config>
   </edit-config>
</rpc>
```

4.9 Read Port Priority Regeneration Table

The following configuration example shows how to read the current state of the Priority Regeneration Table parameter for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <get-config>
      <source>
         <running/>
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <port>
```

```
            <number>1</number>
            </priority-regeneration-table>
         <port/>
      </top>
   </filter>
 </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
 <data>
   <top xmlns=”http://.../config”>
      <port>
         <number>1</number>
         <priority-regeneration-table>
            <priority0>0</priority0>
            <priority1>1</priority1>
            <priority2>2</priority2>
            <priority3>3</priority3>
            <priority4>4</priority4>
            <priority5>5</priority5>
            <priority6>6</priority6>
            <priority7>7</priority7>
         </priority-regeneration-table>
      </port>
   </top>
 </data>
</rpc-reply>
```

4.10 Set Port Priority Regeneration Table

The following configuration example shows how to set the current state of the Priority Regeneration Table parameter for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
 <edit-config>
   <target>
      <running/>
   </target>
   <config>
      <top xmlns=”http://.../config”>
         <bridge>
            <port>
                <number>1</number>
                <priority-regeneration-table>
                       <priority0>0</priority0>
                       <priority1>1</priority1>
                       <priority2>2</priority2>
                       <priority3>3</priority3>
                       <priority4>4</priority4>
                       <priority5>5</priority5>
                       <priority6>6</priority6>
                       <priority7>7</priority7>
                </priority-regeneration-table>
            </port>
         </bridge>
      </top>
   </config>
 </edit-config>
</rpc>
```

4.11 Read Port Priority Code Point Selection

The following configuration example shows how to read which row of the Priority Code Point Encoding Table and Priority Code Point Decoding Table is currently selected for use on a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <port>
              <number>1</number>
              </pcp-selection>
           <port/>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <port>
           <number>1</number>
           <pcp-selection>8P0D</pcp-selection>
        </port>
     </top>
  </data>
</rpc-reply>
```

4.12 Read Port Priority Code Point Decoding Table

The following configuration example shows how to read the current contents of a row in the Priority Code Point Decoding Table for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <port>
              <number>1</number>
              </pcp-decoding-table>
           <port/>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <port>
           <number>1</number>
```

```
            <pcp-decoding-table>
               <type>8P0D</type>
               <table>
                    <pcp0>
                           <priority>0</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp0>
                    <pcp1>
                           <priority>1</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp1>
                    <pcp2>
                           <priority>2</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp2>
                    <pcp3>
                           <priority>3</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp3>
                    <pcp4>
                           <priority>4</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp4>
                    <pcp5>
                           <priority>5</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp5>
                    <pcp6>
                           <priority>6</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp6>
                    <pcp7>
                           <priority>7</priority>
                           <drop-eligible>True</drop-eligible>
                    </pcp7>
               </table>
            </pcp-decoding-table>
         </port>
      </top>
   </data>
</rpc-reply>
```

4.13 Set Port Priority Code Point Encoding Table

The following configuration example shows how to modify the contents of a row in the Priority Code Point Encoding Table for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <bridge>
              <port>
                   <number>1</number>
                   <pcp-encoding-table>
                          <type>8P0D</type>
                          <table>
```

```
                                <priority0>
                                       <pcp-dei-false>0</pcp-dei-false>
                                       <pcp-dei-true>0</pcp-dei-true>
                                </priority0>
                                <priority1>
                                       <pcp-dei-false>1</pcp-dei-false>
                                       <pcp-dei-true>1</pcp-dei-true>
                                </priority1>
                                <priority2>
                                       <pcp-dei-false>2</pcp-dei-false>
                                       <pcp-dei-true>2</pcp-dei-true>
                                </priority2>
                                <priority3>
                                       <pcp-dei-false>3</pcp-dei-false>
                                       <pcp-dei-true>3</pcp-dei-true>
                                </priority3>
                                <priority4>
                                       <pcp-dei-false>4</pcp-dei-false>
                                       <pcp-dei-true>4</pcp-dei-true>
                                </priority4>
                                <priority5>
                                       <pcp-dei-false>5</pcp-dei-false>
                                       <pcp-dei-true>5</pcp-dei-true>
                                </priority5>
                                <priority6>
                                       <pcp-dei-false>6</pcp-dei-false>
                                       <pcp-dei-true>6</pcp-dei-true>
                                </priority6>
                                <priority7>
                                       <pcp-dei-false>7</pcp-dei-false>
                                       <pcp-dei-true>7</pcp-dei-true>
                                </priority7>
                         </table>
                  </pcp-encoding-table>
            </port>
         </bridge>
      </top>
   </config>
  </edit-config>
</rpc>
```

4.14 Set Use_DEI Parameter

The following configuration example shows how to set the current state of the Use_DEI parameter for a specific Bridge Port.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <bridge>
              <port>
                   <number>1</number>
                   <use-dei>False</use-dei>
              </port>
           </bridge>
        </top>
```

```
      </config>
    </edit-config>
</rpc>
```

4.15 Read Required Drop Encoding Parameter

TBD

4.16 Read Service Access Priority Selection

TBD

4.17 Read Service Access Priority Table

TBD

4.18 Read Port Traffic Class Table

TBD

4.19 Set Port Traffic Class Table

TBD

4.20 Read Filtering Database

The following configuration example shows how to obtain general information regarding the Bridge's Filtering Database.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <get-config>
     <source>
        <running/>
     </source>
     <filter type="subtree">
        <top xmlns="http://.../config">
           <filtering-database>
              <information/>
           </filtering-database>

        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <data>
     <top xmlns="http://.../config">
        <filtering-database>
           <information>
              <size>32000</size>
              <static-entries>2000</static-entries>
              <dynamic-entries>30000</dynamic-entries>
              <static-VLAN-registration-entries>2000</static-VLAN-registration-entries>
              <dynamic-VLAN-registration-entries>2000</dynamic-VLAN-registration-entries>
              <age-time>300</age-time>
              <mac-address-registration-entries>2000</mac-address-registration-entries>
           </information>
        </filtering-database>
```

```
      </top>
    </data>
</rpc-reply>
```

4.21 Set Filtering Database Ageing Time

The following configuration example shows how to set the ageing time for Dynamic Filtering Entries.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns="http://.../config">
           <filtering-datbase>
              <information>
                   <aging-time>60</aging-time>
              </information>
           </filtering-database>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.22 Create Static Filtering Entry

The following configuration example shows how to create or update a Static Filtering in the Filtering Database or Permanent Database.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns="http://.../config">
           <filtering-database>
              <filtering-entry operation="create">
                   <database-id>24</database-id>
                   <vid>728</vid>
                   <address>01:02:03:04:00:00</address>
                   <entry-type>static</entry-type>
                   <port-map>
                          <port-number>1</port-number>
                          <map>forward</map>
                   </port-map>
                   <port-map>
                          <port-number>2</port-number>
                          <map>forward</map>
                   </port-map>
                          :
              </filtering-entry>
           </filtering-database>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.23 Read Static Filtering Entry Range

TBD

4.24 Read Dynamic Filtering Entry

The following configuration example shows how to read a Dynamic Filtering in the Filtering Database.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <filtering-database>
              <filtering-entry>
                   <database-id>24</database-id>
                   <vid>728</vid>
                   <address>01:02:03:04:00:00</address>
                   <entry-type>dynamic</entry-type>
              </filtering-entry>
           </filtering-database>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <filtering-database>
           <filtering-entry>
              <database-id>24</database-id>
              <vid>728</vid>
              <address>01:02:03:04:00:00</address>
              <entry-type>dynamic</entry-type>
              <port-map>
                   <port-number>1</port-number>
                   <map>forward</map>
              </port-map>
              <port-map>
                   <port-number>2</port-number>
                   <map>forward</map>
              </port-map>
                   :
           </filtering-entry>
        </filtering-database>
     </top>
  </data>
</rpc-reply>
```

4.25 Create Static VLAN Registration Filtering Entry

TBD

4.26 Delete Static VLAN Registration Filtering Entry

The following configuration example shows how to delete a VALN Registration Entry from the Filtering Database or Permanent Database.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <filtering-database>
              <filtering-entry operation=”delete”>
                   <database-id>24</database-id>
                   <vid>728</vid>
                   <address>01:02:03:04:00:00</address>
                   <entry-type>static</entry-type>
              </filtering-entry>
           </filtering-database>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.27 Read Dynamic VLAN Registration Filtering Entry

TBD

4.28 Read Dynamic VLAN Registration Filtering Entry Range

TBD

4.29 Read Permanent Database

The following configuration example shows how to obtain general information regarding the Bridge’s Permanent Database.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <filtering-database>
              <permanent-database>
                   <information/>
              </permanent-database>
           </filtering-database>

        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
```

```
        <filtering-database>
           <permanent-database>
              <information>
                   <size>2000</size>
                   <static-entries>1000</static-entries>
                   <static-VLAN-registration-entries>500</static-VLAN-registration-entries>
              </information>
           </permanent-database>
        </filtering-database>
     </top>
  </data>
</rpc-reply>
```

4.30 Read Permanent Database Filtering Entry

TBD

4.31 Read Bridge VLAN Configuration

The following configuration example shows how to obtain general VLAN information from a Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <get-config>
     <source>
        <running/>
     </source>
     <filter type="subtree">
        <top xmlns="http://.../config">
           <bridge>
              <bridge-vlan>
                   <vlan/>
              </bridge-vlan>
           </bridge>

        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <data>
     <top xmlns="http://.../config">
        <bridge>
           <bridge-vlan>
              <vlan>
                   <version>2</version>
                   <max>4094</max>
                   <override-default-pvid>True</override-default-pvid>
                   <protocol-template-format>Ethernet</protocol-template-format>
              </vlan>
           </bridge-vlan>
        </bridge>
     </top>
  </data>
</rpc-reply>
```

4.32 Configure PVID and VID Set Values

The following configuration example shows how to configure the PVID and VID Set value(s)

associated with one or more Ports.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <bridge>
              <port>
                  <number>1</number>
                  <pvid>4000</pvid>
              </port>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.33 Configure Acceptable Frame Types Parameters

The following configuration example shows how to configure the Acceptable Frame Types parameter associated with one or more Ports.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns=”http://.../config”>
           <bridge>
              <port>
                  <number>1</number>
                  <acceptable-frame>admit-all-frames</acceptable-frame>
              </port>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.34 Configure Enable Ingress Filtering Parameters

TBD

4.35 Configure Restricted_VLAN_Registration Parameters

TBD

4.36 Configure Protocol Group Database

TBD

4.37 Configure VLAN Learning Constraints

4.37.1 Read VLAN Learning Constraints

The following configuration example shows how to read the contents of a range of one or more entries in the VLAN Learning Constraints table.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
     </source>
     <filter type=”subtree”>
        <top xmlns=”http://.../config”>
           <bridge>
              <bridged-vlan>
                   <vlan-learning-constraints>
                          <entry>1</entry>
                   </vlan-learning-constraints>
              </bridged-vlan>
           </bridge>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <data>
     <top xmlns=”http://.../config”>
        <bridge>
           <bridge-vlan>
              <vlan-learning-constraints>
                   <entry>1</entry>
                   <type>I</type>
                   <value>400</value>
              </vlan-learning-constraints>
           </bridge-vlan>
        </bridge>
     </top>
  </data>
</rpc-reply>
```

4.37.2 Read VLAN Learning Constraints for VID

TBD

4.37.3 Set VLAN Learning Constraints

TBD

4.37.4 Read VID to FID Allocations

The following configuration example shows how to read the contents of a range of one or more entries in the VID to FID allocation table.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
  <get-config>
     <source>
        <running/>
```

```
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <bridge>
               <bridged-vlan>
                    <vlan-learning-constraints>
                           <vid-to-fid-allocation>
                                  <vid>400</vid>
                                  <type/>
                                  <fid/>
                           </vid-to-fid-allocation>
                    </vlan-learning-constraints>
               </bridged-vlan>
            </bridge>
         </top>
      </filter>
   </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <data>
      <top xmlns=”http://.../config”>
         <bridge>
            <bridge-vlan>
               <vlan-learning-constraints>
                    <vid-to-fid-allocation>
                           <vid>400</vid>
                           <type>fixed</type>
                           <fid>2400</fid>
                    </vid-to-fid-allocation>
               </vlan-learning-constraints>
            </bridge-vlan>
         </bridge>
      </top>
   </data>
</rpc-reply>
```

4.37.5 Read FID Allocation for VID

The following configuration example shows how to read the FID to which a specified VID is currently allocated.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-bridge”>
   <get-config>
      <source>
         <running/>
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <bridge>
               <bridged-vlan>
                    <vlan-learning-constraints>
                           <fid-allocation-for-vid>
                                  <vid>400</vid>
                                  <VID-FID-allocation/>
                                  <fid/>
                           </fid-allocation-for-vid>
                    </vlan-learning-constraints>
               </bridged-vlan>
            </bridge>
```

```
        </top>
      </filter>
  </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <data>
     <top xmlns="http://.../config">
        <bridge>
           <bridge-vlan>
              <vlan-learning-constraints>
                   <fid-allocation-for-vid>
                          <vid>400</vid>
                          <VID-FID-allocation>fixed</VID-FID-allocation>
                          <fid>2400</fid>
                   </fid-allocation-for-vid>
              </vlan-learning-constraints>
           </bridge-vlan>
        </bridge>
     </top>
  </data>
</rpc-reply>
```

4.37.6 Read VIDs Allocation to FID

The following configuration example shows how to read all VIDs currently allocated to a given FID.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <get-config>
     <source>
        <running/>
     </source>
     <filter type="subtree">
        <top xmlns="http://.../config">
           <bridge>
              <bridged-vlan>
                   <vlan-learning-constraints>
                          <vid-allocation-for-fid>
                                 <fid>2400</fid>
                                 <vid-list/>
                          </fid-allocation-for-fid>
                   </vlan-learning-constraints>
              </bridged-vlan>
           </bridge>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-bridge">
  <data>
     <top xmlns="http://.../config">
        <bridge>
           <bridge-vlan>
              <vlan-learning-constraints>
                   <vid-allocation-for-fid>
                          <fid>2400</fid>
                          <vid-list>
                                 <vid>400</vid>
                                 <type>fixed</type>
```

```
                        </vid-list>
                        <vid-list>
                                <vid>401</vid>
                                <type>fixed</type>
                        </vid-list>
                                :
                    </vid-allocation-for-fid>
                </vlan-learning-constraints>
            </bridge-vlan>
        </bridge>
    </top>
  </data>
</rpc-reply>
```

4.37.7 Set VID to FID Allocation

TBD

4.37.8 Delete VID to FID Allocation

TBD

4.38 Provider Bridge

Provider Bridges interconnect the separate MACs of the IEEE 802 LANs that compose a Provider Bridged Network, relaying frames to provide connectivity between all LANs that provide customer interfaces for each service instance.

Figure 2 below illustrated the different port types associated with a Provider Bridge.

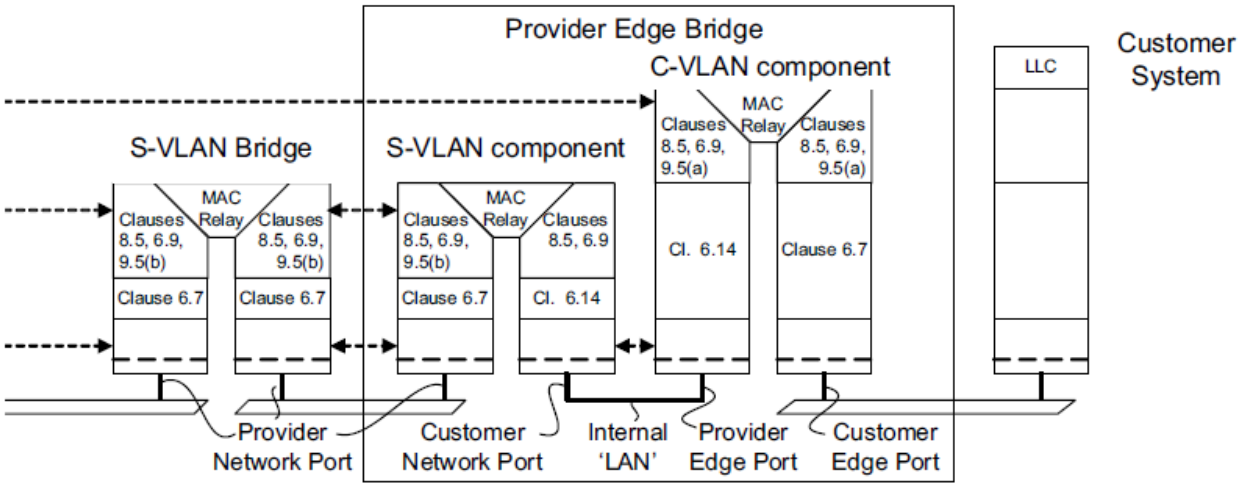

Figure 2: Provider Bridge Decomposition

The internal ports, LANs, and C-VLAN components of a Provider Edge Bridge are not managed directly using the managed objects (defined in IEEE 802.1Q-2011 clause 12.4 through 12.12).

4.38.1 Read Provider Bridge Port Type

The following configuration example shows how to obtain information regarding the designated type

of an externally accessible port on a Provide Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <get-config>
     <source>
        <running/>
     </source>
     <filter type="subtree">
        <top xmlns="http://.../config">
           <port>
              <number>1</number>
              <pb-type/>
           <port/>
        </top>
     </filter>
  </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <data>
     <top xmlns="http://.../config">
        <port>
           <number>1</number>
           <pb-type>customer-edge</pb-type>
        </port>
     </top>
  </data>
</rpc-reply>
```

4.38.2 Configure Provider Bridge Port Type

The following configuration example shows how to designate the type of an externally accessible port on a Provider Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns="http://.../config">
           <bridge>
              <port>
                  <number>2</number>
                  <pb-type>provider-network</pb-type>
              </port>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.38.3 Read VID Translation Table Entry

The following configuration example shows how to read the S-VID associated with the local S-VID.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <get-config>
```

```
      <source>
         <running/>
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <port>
               <number>1</number>
               <network-port-VID-translation-table-entry>
                    <local-vid>400</local-vid>
               </network-port-VID-translation-table-entry>
            <port/>
         </top>
      </filter>
   </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-PB”>
   <data>
      <top xmlns=”http://.../config”>
         <port>
            <number>1</number>
            <network-port-VID-translation-table-entry>
               <local-vid>400</local-vid>
               <relay-vid>3400</relay-vid>
            </network-port-VID-translation-table-entry>
         </port>
      </top>
   </data>
</rpc-reply>
```

4.38.4 Read C-VID Registration Table Entry

The following configuration example shows how to read the VLAN Identifier of the service associated with a specific Customer VLAN in the C-VLAN component of a Provider Edge Bridge.

```
<rpc message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-PB”>
   <get-config>
      <source>
         <running/>
      </source>
      <filter type=”subtree”>
         <top xmlns=”http://.../config”>
            <port>
               <number>1</number>
               <customer-edge-port-cvid-registration-table>
                    <cvid>400</cvid>
               </customer-edge-port-cvid-registration-table>
            <port/>
         </top>
      </filter>
   </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns=" ieee:ns:yang:ieee-dot1Q-PB”>
   <data>
      <top xmlns=”http://.../config”>
         <port>
            <number>1</number>
            <customer-edge-port-cvid-registration-table>
               <cvid>400</cvid>
```

```
            <svid>3400</svid>
            <untagged-PEP>False</untagged-PEP>
            <untagged-CEP>False</untagged-CEP>
          </customer-edge-port-cvid-registration-table>
        </port>
      </top>
    </data>
</rpc-reply>
```

4.38.5 Set Provider Edge Port Configuration

The following configuration example shows how to modify the configuration of a Provider Edge Port in the C-VLAN component of a Provider Edge Bridge.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <edit-config>
     <target>
        <running/>
     </target>
     <config>
        <top xmlns="http://.../config">
           <bridge>
              <port>
                   <number>2</number>
                   <provider-edge-port>
                          <svid>3400</svid>
                   <provider-edge-port>
                   <pvid>4</pvid>
                   <default-priority>0</default-priority>
                   <acceptable-frame>admit-only-VLAN-tagged-frames</acceptable-frame>
                   <ingress-filtering>enabled</ingress-filtering>
              </port>
           </bridge>
        </top>
     </config>
  </edit-config>
</rpc>
```

4.38.6 Read Service Priority Regeneration Table

The following configuration example shows how to read the current contents of the Priority Regeneration Table for an internal Customer Network Port connected to the C-VLAN component associated with a Customer Edge Port.

```
<rpc message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <get-config>
     <source>
        <running/>
     </source>
     <filter type="subtree">
        <top xmlns="http://.../config">
           <port>
              <number>1</number>
              <service-priority-regeneration-table>
                   <svid>3400</svid>
              </service-priority-regeneration-table>
           <port/>
        </top>
```

```
      </filter>
    </get-config>
</rpc>

<rpc-reply message-id="101" xmlns=" ieee:ns:yang:ieee-dot1Q-PB">
  <data>
     <top xmlns="http://.../config">
        <port>
           <number>1</number>
           <priority-regeneration-table>
              <priority0>0</priority0>
              <priority1>1</priority1>
              <priority2>2</priority2>
              <priority3>3</priority3>
              <priority4>4</priority4>
              <priority5>5</priority5>
              <priority6>6</priority6>
              <priority7>7</priority7>
           </priority-regeneration-table>
        </port>
     </top>
  </data>
</rpc-reply>
```

To be completed.

5 Security Considerations

The YANG module defined in this document is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

6 Outstanding Specification Areas

The following items are outstanding and need to be addressed, moving forward:

1. Should the YANG configuration model be based on an augmentation of the (existing) YANG interface management YANG data model [RFC7223] or based upon the identification of a port that may participate in 802.1Q configuration?
2. General rationalization of the YANG model structure.
3. Completion of the YANG 802.1Q type definitions.
4. Should we consider deviation from the existing specified 802.1Q-2011 Bridge management objects (as defined in Clause 12), in support of the YANG configuration model specifications?
5. Determine default settings for configurable parameters.

6. Determine mandatory configurable parameters.
7. Complete example configuration examples.

To be completed.

This page intentionally left (mostly) blank.