```
BESS Working Group                                          H. Shah, Ed.
Internet-Draft                                         Ciena Corporation
Intended status: Standards Track                     P. Brissette, Ed.
Expires: January 3, 2020                             Cisco Systems, Inc.
                                                            I. Chen, Ed.
                                                   The MITRE Corporation
                                                         I. Hussain, Ed.
                                                   Infinera Corporation
                                                             B. Wen, Ed.
                                                                 Comcast
                                                    K. Tiruveedhula, Ed.
                                                        Juniper Networks
                                                           July 02, 2019
```

YANG Data Model for MPLS-based L2VPN
draft-ietf-bess-l2vpn-yang-10.txt

Abstract

This document describes a YANG data model for Layer 2 VPN (L2VPN) services over MPLS networks. These services include point-to-point Virtual Private Wire Service (VPWS) and multipoint Virtual Private LAN service (VPLS) that uses LDP and BGP signaled Pseudowires. It is expected that this model will be used by the management tools run by the network operators in order to manage and monitor the network resources that they use to deliver L2VPN services.

This document also describes the YANG data model for the Pseudowires. The independent definition of the Pseudowires facilitates its use in Ethernet Segment and EVPN data models defined in separate document.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is a network management protocol that defines mechanisms to manage network devices. YANG [RFC7950] is a modular language that represents data structures in an XML or JSON tree format, and is used as a data modeling language for the NETCONF.

This document defines a YANG data model for MPLS based Layer 2 VPN services (L2VPN) [RFC4664] and includes switching between the local attachment circuits. The L2VPN model covers point-to-point VPWS and Multipoint VPLS services. These services use signaling of Pseudowires across MPLS networks using LDP [RFC8077][RFC4762] or BGP[RFC4761].

The data model covers Ethernet based Layer 2 services. The Ethernet Attachment Circuits are not defined. Instead, they are leveraged from other standards organizations such as IEEE802.1 and Metro Ethernet Forum (MEF).

Other Layer 2 services, such as ATM, Frame Relay, TDM, etc are included in the scope but will be covered as the future work items.

The objective of the model is to define building blocks that can easily be assembled in different order to realize different services.

The data model uses following constructs for configuration and management:

o Configuration

o Operational State

o Executables (Actions)

o Notifications

This document focuses on definition of configuration, state and notification objects.

The L2VPN data object model uses the instance centric approach. The L2VPN instance is recognized by network instance model. The network-instance container is defined in network instance model [I-D.ietf-netmod-ni-model].

Within this network instance, L2VPN container contains definitions of a set of common parameters, a list of PWs and a list of endpoints. A special constraint is added for the VPWS configuration such that only two endpoints are allowed in the list of endpoints.

The Pseudowire data object model is defined independent of the L2VPN data object model to allow its inclusion in the Ethernet Segment and EVPN data objects.

The L2VPN data object model augments Psuedowire data object for its definition.

The document also includes Notifications used by the L2VPN object model

2. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. L2VPN YANG Model

3.1. Overview

The document defines configuration of one single container for L2VPN. Within the l2vpn container, common parameters and a list of endpoints are defined. For the point-to-point VPWS configuration, endpoint list is used with the constraint that limits the number of endpoints to be two. For the multipoint service, endpoint list is used. Each endpoint contains the common definition that is either an attachment circuit, a pseudowire or a redundancy group. The previous versions of this document represented VPWS service with definition of endpoint-a and endpoint-z while VPLS with a list of endpoints. This duplicattion is removed with simplified version whereby list of endpoints is used for both. When defining VPWS, the numnber of endpoints is constrained to two endpoints.

The l2vpn container also includes definition of common building blocks for redundancy-grp templates and pseudowire-templates.

The State objects have been consolidated with the configuration object as per the recommendations provided by the Guidelines for Yang Module Authors document.

The IETF working group has defined the VPWS and VPLS services that leverages the pseudowire technologies defined by the PWE3 working group. A large number of RFCs from these working groups cover this subject matter. Hence, it is prudent that this document state the scope of the MPLS L2VPN object model definitions.

The following documents are within the scope. This is not an exhaustive list but a representation of documents that are covered for this work:

- o Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3) [RFC3916]
- o Pseudo-wire Emulation Edge-to-Edge (PWE3) Architecture [RFC3985]
- o IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3) [RFC4446]
- o Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP) [RFC8077]
- o Encapsulation Methods for Transport of Ethernet over MPLS Networks [RFC4448]
- o Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN [RFC4385]
- o Requirements for Multi-Segment Pseudowire Emulation Edge-to-Edge (PWE3) [RFC5254]
- o An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge [RFC5659]
- o Segmented Pseudowire [RFC6073]
- o Framework for Layer 2 Virtual Private Networks [RFC4664]
- o Service Requirements for Layer 2 Provider-Provisioned Virtual Private Networks [RFC4665]
- o Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling [RFC4761]
- o Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling [RFC4762]
- o Attachment Individual Identifier (AII) Types for Aggregation [RFC5003]
- o Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs) [RFC6074]
- o Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network [RFC6391]

- o Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling [RFC6624]
- o Extensions to the Virtual Private LAN Service (VPLS) Provider Edge (PE) Model for Provider Backbone Bridging [RFC7041]
- o LDP Extensions for Optimized MAC Address Withdrawal in a Hierarchical Virtual Private LAN Service (H-VPLS) [RFC7361]
- o Using the generic associated channel label for Pseudowire in the MPLS Transport Profile [RFC6423]
- o Pseudowire status for static pseudowire [RFC6478]

The specifics of pseudowire over MPLS-TP LSPs is in scope. However, the initial effort addresses definitions of object models that are commonly deployed.

The IETF work in L2VPN and PWE3 working group relating to L2TP, OAM, multicast (e.g. p2mp, etree, etc) and access specific protocols such as G.8032, MSTP, etc is out-of-scope for this document.

The following is the high level view of the L2VPN data model.

```
 PW // Container
         PW specific attributes

         PW template definition


 template-ref Redundancy-Group // redundancy-group
                    template
                    attributes

 Network Instance // container
    l2vpn  // containter

         common attributes


         BGP-parameters // container
                           common attributes
                           auto-discovery attributes
                           signaling attributes

         // list of PWs being used
         PW // container
                 template-ref PW
                 attribute-override

         PBB-parameters // container
                           pbb specific attributes

         VPWS-constraints // rule to limit number of endpoints to two

         // List of endpoints, where each member endpoint container is -
                 PW // reference
                 redundancy-grp // container
                         AC // eventual reference to standard AC
                         PW // reference
```

Figure 1

3.2. Latest addition

Pseudowire module is extended to include,

Multi-segment PW - a new attribute is added to pseudowire that identifies the pseudowire as a member of the multi-segment pseudowire. Two pseudowire members in a VPWS, configures a multi-segment pseudowire at the switching PE.

Pseudowire load-balancing - The load-balancing behaviour for a pseudowire can be configured either using the FAT label that resides below the pseudowire label or Entropy label with Entropy label indicator above the pseudowire label. By default, the load-balancing is disabled.

FEC 129 related - AGI, SAII and TAII string configurations is added to faciliate FEC 129 based pseudowire configuration.

3.3. Open issues and next steps

This section provides updates on open issues and will be removed before publication. Authors believes the document has covered the topics within the scope of the document. However, there are items, such as PW Headend, VPLS IRB, etc that can be candidate for inclusion. The authors would like to progress the document to publication for general availability with current content and tackle the other topics in a follow up document.

3.4. Pseudowire Common

3.4.1. Pseudowire

Pseudowire definitions is moved to a seperate container in order to allow Ethernet Segment and EVPN models can refer without having to pull down L2VPN container.

3.4.2. pw-templates

The pw-templates container contains a list of pw-template. Each pw-template defines a list of common pseudowire attributes such as PW MTU, control word support etc.

3.5. L2VPN Common

3.5.1. redundancy-group-templates

The redundancy-group-template contains a list of templates. Each template defines common attributes related to redundancy such as protection mode, reversion parameters, etc.

3.6. L2VPN instance

The network instance container defined in the network instance model [I-D.ietf-rtgwg-ni-model] identifies the L2VPN instance. One of the value defined by the ni-type used in the instance model refers to VSI (Virtual Switch Instance) to denote the L2VPN instance. The name attribute field is used as the key to refer to specific network instance. Network Instance of type VSI anchors L2VPN container with a list of endpoints which when limited to two entries represents point to point service (i.e. VPWS) while more than two endpoints represent multipoint service (i.e. VPLS). Within a service instance, a set of common attributes are defined, followed by a list of PWs and a list of endpoints.

3.6.1. common attributes

The common attributes apply to entire L2VPN instance. These attributes typically include attributes such as mac-aging-timer, BGP related parameters (if using BGP signaling), discovery-type, etc.

3.6.2. PW list

The PW list is the number of PWs that are being used for a given L2VPN instance. Each PW entry refers to PW template to inherit common attributes for the PW. The one or more attributes from the template can be overriden. It further extends definitions of more PW specific attributes such as use of control word, mac withdraw, what type of signaling (i.e. LDP or BGP), setting of the TTL, etc.

3.6.3. List of endpoints

The list of endpoints define the characteristics of the L2VPN service. In the case of VPWS, the list is limited to two entries while for VPLS, there could be many.

Each entry in the endpoint list, may hold AC, PW or redundancy-grp references. The core aspect of endpoint container is its flexible personality based on what user decides to include in it. It is future-proofed with possible extensions that can be included in the endpoint container such as Integrated Route Bridging (IRB), PW Headend, Virtual Switch Instance, etc.

The endpoint entry also includes the split-horizon attribute which defines the frame forwarding restrictions between the endpoints belonging to same split-horizon group. This construct permits multiple instances of split horizon groups with its own endpoint members. The frame forwarding restrictions does not apply between endpoints that belong to two different split horizon groups.

3.6.3.1. ac

Attachment Circuit (AC)resides within endpoint entry either as an independent entity or as a member of the redundancy group. AC is not defined in this document but references the definitions specified by other working groups and standard bodies.

3.6.3.2. pw

The Pseudo-wire resides within endpoint entry either as an independent entity or as a member of the redundancy group. The PW refers to one of the entry in the list of PWs defined with the L2VPN instance.

3.6.3.3. redundancy-grp choice

The redundancy-grp is a generic redundancy construct which can hold primary and backup members of AC and PWs. This flexibility permits combinations of -

o primary and backup AC

o primary and backup PW

o primary AC and backup PW

o primary PW and backup AC

The redundancy group also defines attributes of the type of redundancy, such as protection mode, reroute mode, reversion related parameters, etc.

3.6.4. point-to-point or multipoint service

The point-to-point service as defined for VPWS is represented by a list of endpoints and is limited to two entries by the VPWS constrain rules

The multipoint service as defined for VPLS is represented by a list of endpoints.

The list of endpoints with one entry is invalid.

The augmentation of ietf-l2vpn module is TBD. All IP addresses defined in this module are currently scoped under global VRF/table.

3.6.5. multi-segment pseudowire

The multi-segment pseudowire is expressed as configuration of two pseudowire segments at the switching PEs that provides end-to-end PW path between two terminating PEs consisting of multiple pseudowire segments.

The multi-segment pseudowire is configured at switching PE using two endpoints that consists of pseudowires of type "ms-pw-members". The VPWS service construct is used with "vpws constraint" that restricts the number of endpoints to two.

To verify consistency, a) verify that both endpoints are using ms-pw-member pseudowires and b) it is only used as for VPWS configuration at the switching PE.

3.7. Operational State

The operational state of L2VPN attributes has been consolidated with the configuration as per recommendations from the guidelines for the YANG author document.

3.8. Yang tree

```
module: ietf-pseudowires
  +--rw pseudowires
     +--rw pseudowire* [name]
     |  +--rw name                     string
     |  +--ro state?                   pseudowire-status-type
     |  +--rw template?                pw-template-ref
     |  +--rw mtu?                     uint16
     |  +--rw mac-withdraw?            boolean
     |  +--rw pw-loadbalance?          enumeration
     |  +--rw ms-pw-member?            boolean
     |  +--rw cw-negotiation?          cw-negotiation-type
     |  +--rw tunnel-policy?           string
     |  +--rw (pw-type)?
     |     +--:(configured-pw)
     |     |  +--rw peer-ip?           inet:ip-address
     |     |  +--rw pw-id?             uint32
     |     |  +--rw group-id?          uint32
     |     |  +--rw icb?               boolean
     |     |  +--rw transmit-label?    rt-types:mpls-label
     |     |  +--rw receive-label?     rt-types:mpls-label
     |     |  +--rw generalized?       boolean
     |     |  +--rw agi?               string
     |     |  +--rw saii?              string
```

```
      |     |  +--rw taii?             string
      |     +--:(bgp-pw)
      |     |  +--rw remote-pe-id?     inet:ip-address
      |     +--:(bgp-ad-pw)
      |        +--rw remote-ve-id?     uint16
     +--rw pw-templates
        +--rw pw-template* [name]
           +--rw name              string
           +--rw mtu?              uint16
           +--rw cw-negotiation?   cw-negotiation-type
           +--rw tunnel-policy?    string

module: ietf-l2vpn
  +--rw l2vpn
     +--rw redundancy-group-templates
        +--rw redundancy-group-template* [name]
           +--rw name               string
           +--rw protection-mode?   enumeration
           +--rw reroute-mode?      enumeration
           +--rw dual-receive?      boolean
           +--rw revert?            boolean
           +--rw reroute-delay?     uint16
           +--rw revert-delay?      uint16

  augment /ni:network-instances/ni:network-instance/ni:ni-type:
    +--:(l2vpn)
       +--rw type?               identityref
       +--rw mtu?                uint16
       +--rw mac-aging-timer?    uint32
       +--rw service-type?       l2vpn-service-type
       +--rw discovery-type?     l2vpn-discovery-type
       +--rw signaling-type      l2vpn-signaling-type
       +--rw bgp-parameters
       |  +--rw vpn-id?   string
       |  +--rw rd-rt
       |     +--rw route-distinguisher?   rt-types:route-distinguisher
       |     +--rw vpn-target* [route-target]
       |        +--rw route-target         rt-types:route-target
       |        +--rw route-target-type    rt-types:route-target-type
       +--rw bgp-signaling
       |  +--rw site-id?      uint16
       |  +--rw site-range?   uint16
       +--rw endpoint* [name]
       |  +--rw name                          string
       |  +--rw (ac-or-pw-or-redundancy-grp)?
       |  |  +--:(ac)
       |  |  |  +--rw ac* [name]
       |  |  |     +--rw name     if:interface-ref
       |  |  |     +--ro state?   operational-state-type
       |  |  +--:(pw)
       |  |  |  +--rw pw* [name]
       |  |  |     +--rw name     pw:pseudowire-ref
       |  |  |     +--ro state?   -> /pw:pseudowires/pseudowire[pw:name=current(
)/../name]/state
       |  |  +--:(redundancy-grp)
       |  |     +--rw (primary)
       |  |     |  +--:(primary-ac)
       |  |     |  |  +--rw primary-ac
       |  |     |  |     +--rw name?    if:interface-ref
       |  |     |  |     +--ro state?   operational-state-type
       |  |     |  +--:(primary-pw)
       |  |     |     +--rw primary-pw* [name]
       |  |     |        +--rw name     pw:pseudowire-ref
       |  |     |        +--ro state?   -> /pw:pseudowires/pseudowire[pw:name=cu
rrent()/../name]/state
       |  |     +--rw (backup)?
       |  |     |  +--:(backup-ac)
       |  |     |  |  +--rw backup-ac
       |  |     |  |     +--rw name?    if:interface-ref
       |  |     |  |     +--ro state?   operational-state-type
       |  |     |  +--:(backup-pw)
       |  |     |     +--rw backup-pw* [name]
       |  |     |        +--rw name          pw:pseudowire-ref
       |  |     |        +--ro state?        -> /pw:pseudowires/pseudowire[pw:na
me=current()/../name]/state
       |  |     |        +--rw precedence?   uint32
       |  |     +--rw template?           redundancy-group-template-ref
       |  |     +--rw protection-mode?    enumeration
       |  |     +--rw reroute-mode?       enumeration
       |  |     +--rw dual-receive?       boolean
       |  |     +--rw revert?             boolean
       |  |     +--rw reroute-delay?      uint16
       |  |     +--rw revert-delay?       uint16
       |  +--rw split-horizon-group?      string
       +--rw vpws-constraints
       +--rw pbb-parameters
          +--rw (component-type)?
             +--:(i-component)
             |  +--rw i-sid?                    i-sid-type
             |  +--rw backbone-src-mac?         yang:mac-address
             +--:(b-component)
                +--rw bind-b-component-name?   l2vpn-instance-name-ref
                +--ro bind-b-component-type?   identityref
  augment /pw:pseudowires/pw:pseudowire:
    +--rw vccv-ability?     boolean
    +--rw request-vlanid?   uint16
    +--rw vlan-tpid?        string
    +--rw ttl?              uint8
  augment /pw:pseudowires/pw:pseudowire/pw:pw-type:
    +--:(bgp-pw)
    |  +--rw bgp-pw
    |     +--rw remote-pe-id?   inet:ip-address
    +--:(bgp-ad-pw)
       +--rw bgp-ad-pw
          +--rw remote-ve-id?   uint16

  notifications:
    +---n l2vpn-state-change-notification
       +--ro l2vpn-instance-name?      l2vpn-instance-name-ref
       +--ro l2vpn-instance-type?      -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:type
       +--ro endpoint?                 -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint/name
       +--ro (ac-or-pw-or-redundancy-grp)?
       |  +--:(ac)
       |  |  +--ro ac?                 -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/ac/name
       |  +--:(pw)
       |  |  +--ro pw?                 -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/pw/name
       |  +--:(redundancy-grp)
       |     +--ro (primary)
       |     |  +--:(primary-ac)
       |     |  |  +--ro primary-ac?   -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/primary-ac/name
       |     |  +--:(primary-pw)
       |     |     +--ro primary-pw?   -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/primary-pw/name
       |     +--ro (backup)?
       |        +--:(backup-ac)
       |        |  +--ro backup-ac?    -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/backup-ac/name
       |        +--:(backup-pw)
       |           +--ro backup-pw?    -> /ni:network-instances/network-instance
[ni:name=current()/../l2vpn-instance-name]/l2vpn:endpoint[l2vpn:name=current()/.
./endpoint]/backup-pw/name
       +--ro state?                    identityref
```

Figure 2

4. YANG Module

The L2VPN configuration container is logically divided into following high level config areas:

```
<CODE BEGINS> file "ietf-pseudowires@2018-10-17.yang"
 module ietf-pseudowires {
  namespace "urn:ietf:params:xml:ns:yang:ietf-pseudowires";
  prefix "pw";

  import ietf-inet-types {
    prefix "inet";
```

```
  }

  import ietf-routing-types {
    prefix "rt-types";
  }

  organization  "ietf";
  contact       "ietf";
  description   "Pseudowire YANG model";

  revision "2018-10-17" {
    description "Second revision " +
                "  - Added group-id and attachment identifiers " +
                "";
    reference   "";
  }

  revision "2017-06-26" {
    description "Initial revision " +
                "  - Created a new model for pseudowires, which used " +
                "    to be defined within the L2VPN model " +
                "";
    reference   "";
  }

  /* Typedefs */

  typedef pseudowire-ref {
    type leafref {
      path "/pw:pseudowires/pw:pseudowire/pw:name";
    }
    description "A type that is a reference to a pseudowire";
  }

  typedef pw-template-ref {
    type leafref {
      path "/pw:pseudowires/pw:pw-templates/pw:pw-template/pw:name";
    }
    description "A type that is a reference to a pw-template";
  }

  typedef cw-negotiation-type {
    type enumeration {
      enum "non-preferred" {
        description "No preference for control-word";
      }
      enum "preferred" {
        description "Prefer to have control-word negotiation";
      }
    }
    description "control-word negotiation preference type";
  }

  typedef pseudowire-status-type {
    type bits {
      bit pseudowire-forwarding {
        position 0;
        description "Pseudowire is forwarding";
      }
      bit pseudowire-not-forwarding {
        position 1;
        description "Pseudowire is not forwarding";
      }
      bit local-attachment-circuit-receive-fault {
        position 2;
        description "Local attachment circuit (ingress) receive " +
                    "fault";
      }
      bit local-attachment-circuit-transmit-fault {
        position 3;
        description "Local attachment circuit (egress) transmit " +
                    "fault";
      }
      bit local-PSN-facing-PW-receive-fault {
        position 4;
        description "Local PSN-facing PW (ingress) receive fault";
      }
      bit local-PSN-facing-PW-transmit-fault {
        position 5;
        description "Local PSN-facing PW (egress) transmit fault";
      }
      bit PW-preferential-forwarding-status {
        position 6;
        description "Pseudowire preferential forwarding status";
      }
      bit PW-request-switchover-status {
        position 7;
        description "Pseudowire request switchover status";
      }
    }
    description
      "Pseudowire status type, as registered in the IANA " +
      "Pseudowire Status Code Registry";
  }

  /* Data */
  container pseudowires {
    description "Configuration management of pseudowires";
    list pseudowire {
      key "name";
      description "A pseudowire";
      leaf name {
        type string;
        description "pseudowire name";
      }
      leaf state {
        type pseudowire-status-type;
        config false;
        description "pseudowire operation status";
        reference "RFC 4446 and IANA Pseudowire Status Codes " +
                  "Registery";
      }
      leaf template {
        type pw-template-ref;
        description "pseudowire template";
      }
      leaf mtu {
        type uint16;
        description "PW MTU";
      }
      leaf mac-withdraw {
        type boolean;
        default false;
        description "Enable (true) or disable (false) MAC withdraw";
      }
      leaf pw-loadbalance {
        type enumeration {
          enum "disabled" {
            value 0;
            description "load-balancing disabled";
          }
          enum "fat-pw" {
            value 1;
            description "load-balance using FAT label below PW label";
          }
          enum "entropy" {
            value 2;
            description "load-balance using ELI/EL above PW label";
          }
        }
        description "PW load-balancing";
      }
      leaf ms-pw-member {
        type boolean;
        default false;
        description "Enable (true) or disable (false) not a member of MS-PW";
      }
      leaf cw-negotiation {
        type cw-negotiation-type;
        description "cw-negotiation";
      }
      leaf tunnel-policy {
        type string;
        description "tunnel policy name";
      }
      choice pw-type {
        description "A choice of pseudowire type";
        case configured-pw {
          leaf peer-ip {
            type inet:ip-address;
            description "peer IP address";
          }
          leaf pw-id {
            type uint32;
            description "pseudowire id";
          }
          leaf group-id {
            type uint32;
            description "group id";
          }
          leaf icb {
            type boolean;
            description "inter-chassis backup";
          }
          leaf transmit-label {
            type rt-types:mpls-label;
            description "transmit lable";
          }
          leaf receive-label {
            type rt-types:mpls-label;
            description "receive label";
          }
          leaf generalized {
            type boolean;
            description "generalized pseudowire id FEC element";
          }
          leaf agi {
            type string;
            description "attachment group identifier";
          }
          leaf saii {
            type string;
            description "source attachment individual identifier";
          }
          leaf taii {
            type string;
            description "target attachment individual identifier";
          }
        }
        case bgp-pw {
          leaf remote-pe-id {
            type inet:ip-address;
            description "remote pe id";
          }
        }
        case bgp-ad-pw {
          leaf remote-ve-id {
            type uint16;
            description "remote ve id";
          }
        }
      }
    }
    container pw-templates {
      description "pw-templates";
      list pw-template {
        key "name";
        description "pw-template";
        leaf name {
          type string;
          description "name";
        }
        leaf mtu {
          type uint16;
          description "pseudowire mtu";
        }
        leaf cw-negotiation {
          type cw-negotiation-type;
          default "preferred";
          description
            "control-word negotiation preference";
        }
        leaf tunnel-policy {
          type string;
          description "tunnel policy name";
        }
      }
    }
  }
}
```

```
<CODE ENDS>
<CODE BEGINS> file "ietf-l2vpn@2019-05-28.yang"
module ietf-l2vpn {
  namespace "urn:ietf:params:xml:ns:yang:ietf-l2vpn";
  prefix "l2vpn";

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-routing-types {
    prefix "rt-types";
  }

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-network-instance {
    prefix "ni";
  }

  import ietf-pseudowires {
    prefix "pw";
  }

  organization  "ietf";
  contact       "ietf";
  description   "l2vpn";

  revision "2019-05-28" {
    description "Nineth revision " +
                "  - Used bgp parameters hierarchy common to L2VPN and EVPN " +
                "";
    reference   "";
  }

  revision "2018-02-06" {
    description "Eighth revision " +
                "  - Incorporated ietf-network-instance model " +
                "  - change the type of attachment circuit to " +
                "    if:interface-ref " +
                "";
    reference   "";
  }

  revision "2017-09-21" {
    description "Seventh revision " +
                "  - Fixed yangdump errors " +
                "";
    reference   "";
  }
  revision "2017-06-26" {
    description "Sixth revision " +
                "  - Removed unused module mpls " +
                "  - Renamed l2vpn-instances-state to l2vpn-instances " +
                "  - Added pseudowire status as defined in RFC4446 and " +
                "    IANA Pseudowire Status Codes Register " +
                "  - Added notifications " +
                "  - Moved PW definition out of L2VPN " +
                "  - Moved model to NMDA style specified in " +
                "    draft-dsdt-nmda-guidelines-01.txt " +
                "  - Renamed l2vpn-instances and l2vpn-instance to " +
                "    instances and instance to shorten xpaths " +
                "";
    reference   "";
  }

  revision "2017-03-06" {
    description "Sixth revision " +
                "  - Removed the 'common' container and move pw-templates " +
                "    and redundancy-group-templates up a level " +
                "  - Consolidated the endpoint configuration such that " +
                "    all L2VPN instances has a list of endpoint.  For " +
                "    certain types of L2VPN instances such as VPWS where " +
                "    each L2VPN instance is limited to at most two " +
                "    endpoint, additional augment statements were included " +
                "    to add necessary constraints " +
                "  - Removed discovery-type and signaling-type operational " +
                "    state from VPLS pseudowires, as these two parameters " +
                "    are configured as L2VPN parameters rather than " +
                "    pseudowire paramteres " +
                "  - Renamed l2vpn-instances to l2vpn-instances-state " +
                "    in the operational state branch " +
                "  - Removed BGP parameter groupings and reused " +
                "    ietf-routing-types.yang module instead " +
                "";
    reference "";
  }

  revision "2016-10-24" {
    description "Fifth revision " +
                "  - Edits based on Giles's comments " +
                "    5) Remove relative leafrefs in groupings, " +
                "       and the resulting new groupings are: " +
                "       (a) bgp-auto-discovery-parameters-grp " +
                "       (b) bgp-signaling-parameters-grp " +
                "       (c) endpoint-grp " +
                "    11) Merge VPLS and VPWS into one single list " +
                "       and use augment statements to handle " +
                "       differences between VPLS and VPWS " +
                "  - Add a new grouping l2vpn-common-parameters-grp " +
                "    to make VPLS and VPWS more consistent";
    reference "";
  }

  revision "2016-05-31" {
    description "Fourth revision " +
                "  - Edits based on Giles's comments " +
                "    1) Change enumeration to identityref type for: " +
                "       (a) l2vpn-service-type " +
                "       (b) l2vpn-discovery-type " +
                "       (c) l2vpn-signaling-type " +
                "       bgp-rt-type, cw-negotiation, and " +
                "       pbb-component remain enumerations " +
                "    2) Define i-sid-type for leaf 'i-sid' " +
                "       (which is renamed from 'i-tag') " +
                "    3) Rename 'vpn-targets' to 'vpn-target' " +
                "    4) Import ietf-mpls.yang and reuse the " +
                "       'mpls-label' type defined in ietf-mpls.yang " +
                "       transmit-label and receive-label " +
                "    8) Change endpoint list's key to name " +
                "    9) Changed MTU to type uint16 " +
                "";
    reference "";
  }

  revision "2016-03-07" {
    description "Third revision " +
                "  - Changed the module name to ietf-l2vpn " +
                "  - Merged EVPN into L2VPN " +
                "  - Eliminated the definitions of attachment " +
                "    circuit with the intention to reuse other " +
                "    layer-2 definitions " +
                "  - Added state branch";
    reference "";
  }

  revision "2015-10-08" {
    description "Second revision " +
                "  - Added container vpls-instances " +
                "  - Rearranged groupings and typedefs to be " +
                "    reused across vpls-instance and vpws-instances";
    reference "";
  }

  revision "2015-06-30" {
    description "Initial revision";
    reference   "";
  }

  /* identities */

  identity l2vpn-instance-type {
    description "Base identity from which identities of " +
                "l2vpn service instance types are derived";
  }

  identity vpws-instance-type {
    base l2vpn-instance-type;
    description "This identity represents VPWS instance type";
  }

  identity vpls-instance-type {
    base l2vpn-instance-type;
    description "This identity represents VPLS instance type";
  }

  identity link-discovery-protocol {
    description "Base identiy from which identities describing " +
                "link discovery protocols are derived";
  }

  identity lacp {
    base "link-discovery-protocol";
    description "This identity represents LACP";
  }

  identity lldp {
    base "link-discovery-protocol";
    description "This identity represents LLDP";
  }

  identity bpdu {
    base "link-discovery-protocol";
    description "This identity represens BPDU";
  }
```

```
  identity cpd {
    base "link-discovery-protocol";
    description "This identity represents CPD";
  }

  identity udld {
    base "link-discovery-protocol";
    description "This identity represens UDLD";
  }

  identity l2vpn-service {
    description "Base identity from which identities describing " +
                "L2VPN services are derived";
  }

  identity Ethernet {
    base "l2vpn-service";
    description "This identity represents Ethernet service";
  }

  identity ATM {
    base "l2vpn-service";
    description "This identity represents Asynchronous Transfer " +
                "Mode service";
  }

  identity FR {
    base "l2vpn-service";
    description "This identity represent Frame-Relay service";
  }

  identity TDM {
    base "l2vpn-service";
    description "This identity represent Time Devision " +
                "Multiplexing service";
  }
  identity l2vpn-discovery {
    description "Base identity from which identities describing " +
                "L2VPN discovery protocols are derived";
  }

  identity manual-discovery {
    base "l2vpn-discovery";
    description "Manual configuration of l2vpn service";
  }

  identity bgp-auto-discovery {
    base "l2vpn-discovery";
    description "Border Gateway Protocol (BGP) auto-discovery of " +
                "l2vpn service";
  }

  identity ldp-discovery {
    base "l2vpn-discovery";
    description "Label Distribution Protocol (LDP) discovery of " +
                "l2vpn service";
  }

  identity mixed-discovery {
    base "l2vpn-discovery";
    description "Mixed discovery methods of l2vpn service";
  }

  identity l2vpn-signaling {
    description "Base identity from which identities describing " +
                "L2VPN signaling protocols are derived";
  }

  identity static-configuration {
    base "l2vpn-signaling";
    description "Static configuration of labels (no signaling)";
  }

  identity ldp-signaling {
    base "l2vpn-signaling";
    description "Label Distribution Protocol (LDP) signaling";
  }

  identity bgp-signaling {
    base "l2vpn-signaling";
    description "Border Gateway Protocol (BGP) signaling";
  }

  identity mixed-signaling {
    base "l2vpn-signaling";
    description "Mixed signaling methods";
  }

  identity l2vpn-notification-state {
    description "The base identity on which notification states " +
                "are based";
  }

  identity MAC-limit-reached {
    base "l2vpn-notification-state";
    description "MAC limit is reached";
  }
  identity MAC-limit-cleared {
    base "l2vpn-notification-state";
    description "MAC limit is cleared";
  }

  identity MTU-mismatched {
    base "l2vpn-notification-state";
    description "MAC is mismatched";
  }

  identity MTU-mismatched-cleared {
    base "l2vpn-notification-state";
    description "MAC is mismatch is cleared";
  }

  identity state-changed-to-up {
    base "l2vpn-notification-state";
    description "State is changed to UP";
  }

  identity state-changed-to-down {
    base "l2vpn-notification-state";
    description "State is changed to down";
  }

  identity MAC-move-limit-exceeded {
    base "l2vpn-notification-state";
    description "MAC move limit is exceeded";
  }

  identity MAC-move-limit-exceeded-cleared {
    base "l2vpn-notification-state";
    description "MAC move limit exceeded is cleared";
  }

  identity MAC-flap-detected {
    base "l2vpn-notification-state";
    description "MAC flap detected";
  }

  identity port-disabled-due-to-MAC-flap {
    base "l2vpn-notification-state";
    description "Port disabled due to MAC flap";
  }

  /* typedefs */
  typedef l2vpn-service-type {
    type identityref {
      base "l2vpn-service";
    }
    description "L2VPN service type";
  }

  typedef l2vpn-discovery-type {
    type identityref {
      base "l2vpn-discovery";
    }
    description "L2VPN discovery type";
  }

  typedef l2vpn-signaling-type {
    type identityref {
      base "l2vpn-signaling";
    }
    description "L2VPN signaling type";
  }

  typedef link-discovery-protocol-type {
    type identityref {
      base "link-discovery-protocol";
    }
    description "This type is used to identify " +
                "link discovery protocol";
  }

  typedef pbb-component-type {
    type enumeration {
      enum "b-component" {
        description "Identifies as a b-component";
      }
      enum "i-component" {
        description "Identifies as an i-component";
      }
    }
    description "This type is used to identify " +
                "the type of PBB component";
  }

  typedef redundancy-group-template-ref {
    type leafref {
      path "/l2vpn:l2vpn/l2vpn:redundancy-group-templates" +
           "/l2vpn:redundancy-group-template/l2vpn:name";
    }
    description "redundancy-group-template-ref";
  }
  typedef l2vpn-instance-name-ref {
    type leafref {
      path "/ni:network-instances/ni:network-instance" +
           "/ni:name";
    }
    description "l2vpn-instance-name-ref";
  }

  typedef l2vpn-instance-type-ref {
    type leafref {
      path "/ni:network-instances/ni:network-instance" +
           "/l2vpn:type";
    }
    description "l2vpn-instance-type-ref";
  }

  typedef operational-state-type {
    type enumeration {
      enum 'up' {
        description "Operational state is up";
      }
      enum 'down' {
        description "Operational state is down";
      }
    }
    description "operational-state-type";
  }

  typedef i-sid-type {
    type uint32 {
      range "0..16777216";
    }
    description "I-SID type that is 24-bits. " +
                "This should be moved to ieee-types.yang at " +
                "http://www.ieee802.org/1/files/public/docs2015" +
                "/new-mholness-ieee-types-yang-v01.yang";
  }

  /* groupings */

  grouping pbb-parameters-grp {
    description "PBB parameters grouping";
    container pbb-parameters {
      description "pbb-parameters";
      choice component-type {
        description "PBB component type";
        case i-component {
          leaf i-sid {
            type i-sid-type;
            description "I-SID";
          }
          leaf backbone-src-mac {
            type yang:mac-address;
            description "backbone-src-mac";
          }
        }
        case b-component {
          leaf bind-b-component-name {
            type l2vpn-instance-name-ref;
            must "/ni:network-instances" +
                 "/ni:network-instance[ni:name=current()]" +
                 "/l2vpn:type = 'l2vpn:vpls-instance-type'" {
              description "A b-component must be an L2VPN instance " +
                          "of type vpls-instance-type";
            }
            description "Reference to the associated b-component";
          }
          leaf bind-b-component-type {
            type identityref {
              base l2vpn-instance-type;
            }
            must ". = 'l2vpn:vpls-instance-type'" {
              description "The associated b-component must have " +
                          "type vpls-instance-type";
            }
            config false;
            description "Type of the associated b-component";
          }
        }
      }
    }
  }

  grouping pbb-parameters-state-grp {
    description "PBB parameters grouping";
    container pbb-parameters {
      description "pbb-parameters";
      choice component-type {
        description "PBB component type";
        case i-component {
          leaf i-sid {
            type i-sid-type;
            description "I-SID";
          }
          leaf backbone-src-mac {
            type yang:mac-address;
            description "backbone-src-mac";
          }
        }
        case b-component {
          leaf bind-b-component-name {
            type string;
            description "Name of the associated b-component";
          }
          leaf bind-b-component-type {
            type identityref {
              base l2vpn-instance-type;
            }
            must ". = 'l2vpn:vpls-instance-type'" {
              description "The associated b-component must have " +
                          "type vpls-instance-type";
            }
            description "Type of the associated b-component";
          }
        }
      }
    }
  }

  grouping l2vpn-common-parameters-grp {
    description "L2VPN common parameters";
    leaf type {
      type identityref {
        base l2vpn-instance-type;
      }
      description "Type of L2VPN service instance";
    }
    leaf mtu {
      type uint16;
      description "MTU of L2VPN service";
    }
    leaf mac-aging-timer {
      type uint32;
      description "mac-aging-timer, the duration after which" +
                  "a MAC entry is considered aged out";
    }
    leaf service-type {
      type l2vpn-service-type;
      default Ethernet;
      description "L2VPN service type";
    }
    leaf discovery-type {
      type l2vpn-discovery-type;
      default manual-discovery;
      description "L2VPN service discovery type";
    }
    leaf signaling-type {
      type l2vpn-signaling-type;
      mandatory true;
      description "L2VPN signaling type";
    }
  }
  grouping bgp-signaling-parameters-grp {
    description "BGP parameters for signaling";
    leaf site-id {
      type uint16;
      description "Site ID";
    }
    leaf site-range {
      type uint16;
      description "Site Range";
    }
  }

  grouping redundancy-group-properties-grp {
    description "redundancy-group-properties-grp";
    leaf protection-mode {
      type enumeration {
        enum "frr" {
          value 0;
          description "fast reroute";
        }
        enum "master-slave" {
          value 1;
          description "master-slave";
        }
        enum "independent" {
          value 2;
          description "independent";
        }
      }
      description "protection-mode";
    }
    leaf reroute-mode {
      type enumeration {
        enum "immediate" {
          value 0;
          description "immediate reroute";
        }
        enum "delayed" {
          value 1;
          description "delayed reroute";
        }
        enum "never" {
          value 2;
          description "never reroute";
        }
      }
      description "reroute-mode";
    }
    leaf dual-receive {
      type boolean;
      description
      "allow extra traffic to be carried by backup";
    }
    leaf revert {
      type boolean;
      description "allow forwarding to revert to primary " +
                  "after restoring primary";
    }
    leaf reroute-delay {
      when "../reroute-mode = 'delayed'" {
        description "Specify amount of time to " +
                    "delay reroute only when " +
                    "delayed route is configured";
      }
      type uint16;
      description "amount of time to delay reroute";
    }
    leaf revert-delay {
      when "../revert = 'true'" {
        description "Specify the amount of time to " +
                    "wait to revert to primary " +
                    "only if reversion is configured";
      }
      type uint16;
      description "amount ot time to wait to revert to primary";
    }
  }

  grouping endpoint-grp {
    description "A grouping that defines the structure of " +
                "an endpoint";
    choice ac-or-pw-or-redundancy-grp {
      description "A choice ofattachment circuit or " +
                  "pseudowire or redundancy group";
      case ac {
        description "Attachment circuit(s) as an endpoint";
      }
      case pw {
        description "Pseudowire(s) as an endpoint";
      }
      case redundancy-grp {
        description "Redundancy group as an endpoint";
        choice primary {
          mandatory true;
          description "primary options";
          case primary-ac {
            description "primary-ac";
          }
          case primary-pw {
            description "primary-pw";
          }
        }
        choice backup {
          description "backup options";
          case backup-ac {
            description "backup-ac";
          }
          case backup-pw {
            description "backup-pw";
          }
        }
      }
    }
  }

  /* L2VPN YANG Model */

  container l2vpn {
    description "L2VPN specific data";

    container redundancy-group-templates {
      description "redundancy group templates";
      list redundancy-group-template {
        key "name";
        description "redundancy-group-template";
        leaf name {
          type string;
          description "name";
        }
        uses redundancy-group-properties-grp;
      }
    }
  }

  /* augments */
  augment "/ni:network-instances/ni:network-instance/ni:ni-type" {
    description
      "Augmentation for L2VPN instance";
    case l2vpn {
      description "An L2VPN service instance";
      uses l2vpn-common-parameters-grp;
      container bgp-parameters {
        when "../discovery-type = 'l2vpn:bgp-auto-discovery'" {
          description "Parameters used when discovery type is " +
                      "bgp-auto-discovery";
        }
        description "BGP auto-discovery parameters";
        leaf vpn-id {
          type string;
          description "VPN ID";
        }
        container rd-rt {
          leaf route-distinguisher {
            type rt-types:route-distinguisher;
            description "BGP route distinguisher";
          }
          uses rt-types:vpn-route-targets;
          description "Route distiguisher and " +
                      "corresponding VPN route targets";
        }
      }
      container bgp-signaling {
        when "../signaling-type = 'l2vpn:bgp-signaling'" {
          description "Check signaling type: " +
                      "Can only configure BGP signaling if " +
                      "signaling type is BGP";
        }
        description "BGP signaling parameters";
        uses bgp-signaling-parameters-grp;
      }
      list endpoint {
        key "name";
        description "An endpoint";
        leaf name {
          type string;
          description "endpoint name";
        }
        uses endpoint-grp {
          augment "ac-or-pw-or-redundancy-grp/ac" {
            description "Augment for attachment circuit(s) " +
                        "as an endpoint";
            list ac {
              key "name";
              leaf name {
                type if:interface-ref;
                description "Name of attachment circuit";
              }
              leaf state {
                type operational-state-type;
                config false;
                description "attachment circuit up/down state";
              }
              description "An L2VPN instance's " +
                          "attachment circuit list";
            }
          }
          augment "ac-or-pw-or-redundancy-grp/pw" {
            description "Augment for pseudowire(s) as an endpoint";
            list pw {
              key "name";
              leaf name {
                type pw:pseudowire-ref;
                must "(../../../type = " +
                       " 'l2vpn:vpws-instance-type') or " +
                       "(not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vccv-ability)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /request-vlanid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vlan-tpid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /ttl)))" {
                    description "Only a VPWS PW has parameters " +
                                "vccv-ability, request-vlanid, " +
                                "vlan-tpid, and ttl";
                  }
                description "Pseudowire name";
              }
              leaf state {
                type leafref {
                  path "/pw:pseudowires" +
                       "/pw:pseudowire[pw:name=current()/../name]" +
                       "/pw:state";
                }
                config false;
                description "Pseudowire state";
              }
              description "An L2VPN instance's pseudowire list";
            }
          }
          augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
                  "primary/primary-ac" {
            description "Augment for primary-ac";
            container primary-ac {
              description "Primary AC";
              leaf name {
                type if:interface-ref;
                description "Name of attachment circuit";
              }
              leaf state {
                type operational-state-type;
                config false;
                description "attachment circuit up/down state";
              }
            }
          }
          augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
                  "primary/primary-pw" {
            description "Augment for primary-pw";
            list primary-pw {
              key "name";
              leaf name {
                type pw:pseudowire-ref;
                must "(../../../type = " +
                       " 'l2vpn:vpws-instance-type') or " +
                       "(not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vccv-ability)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /request-vlanid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vlan-tpid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /ttl)))" {
                    description "Only a VPWS PW has parameters " +
                                "vccv-ability, request-vlanid, " +
                                "vlan-tpid, and ttl";
                  }
                description "Pseudowire name";
              }
              leaf state {
                type leafref {
                  path "/pw:pseudowires" +
                       "/pw:pseudowire[pw:name=current()/../name]" +
                       "/pw:state";
                }
                config false;
                description "Pseudowire state";
              }
              description "An L2VPN instance’s pseudowire list";
            }
          }
          augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
                  "backup/backup-ac" {
            description "Augment for backup-ac";
            container backup-ac {
              description "Backup AC";
              leaf name {
                type if:interface-ref;
                description "Name of attachment circuit";
              }
              leaf state {
                type operational-state-type;
                config false;
                description "attachment circuit up/down state";
              }
            }
          }
          augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
                  "backup/backup-pw" {
            description "Augment for backup-pw";
            list backup-pw {
              key "name";
              leaf name {
                type pw:pseudowire-ref;
                must "(../../../type = " +
                       " ’l2vpn:vpws-instance-type’) or " +
                       "(not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vccv-ability)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /request-vlanid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /vlan-tpid)) and " +
                       " not(boolean(/pw:pseudowires" +
                       "     /pw:pseudowire[pw:name = current()]" +
                       "     /ttl)))" {
                  description "Only a VPWS PW has parameters " +
                              "vccv-ability, request-vlanid, " +
                              "vlan-tpid, and ttl";
                  }
                description "Pseudowire name";
              }
              leaf state {
                type leafref {
                  path "/pw:pseudowires" +
                       "/pw:pseudowire[pw:name=current()/../name]" +
                       "/pw:state";
                }
                config false;
                description "Pseudowire state";
              }
              description "A list of backup pseudowires";
            }
          }
          augment "ac-or-pw-or-redundancy-grp/redundancy-grp" {
            description "Augment for redundancy group properties";
            leaf template {
              type redundancy-group-template-ref;
              description "Reference a redundancy group " +
                          "properties template";
            }
            uses redundancy-group-properties-grp;
          }
        }
      }
    }
  }

  augment "/pw:pseudowires/pw:pseudowire" {
    description "Augment for peudowire parameters for " +
                "VPWS pseudowires";
    leaf vccv-ability {
      type boolean;
      description "vccvability";
    }
    leaf request-vlanid {
      type uint16;
      description "request vlanid";
    }
    leaf vlan-tpid {
      type string;
      description "vlan tpid";
    }
    leaf ttl {
      type uint8;
      description "time-to-live";
    }
  }

  augment "/pw:pseudowires/pw:pseudowire/pw:pw-type" {
    description "Additional pseudowire types";
    case bgp-pw {
      container bgp-pw {
        description "BGP pseudowire";
        leaf remote-pe-id {
          type inet:ip-address;
          description "remote pe id";
        }
      }
    }
    case bgp-ad-pw {
      container bgp-ad-pw {
        description "BGP auto-discovery pseudowire";
        leaf remote-ve-id {
          type uint16;
          description "remote ve id";
        }
      }
    }
  }

  augment "/ni:network-instances/ni:network-instance/ni:ni-type" +
          "/l2vpn:l2vpn" {
    when "l2vpn:type = 'l2vpn:vpws-instance-type'" {
      description "Constraints only for VPWS pseudowires";
    }
    description "Augment for VPWS instance";
    container vpws-constraints {
      must "(count(../endpoint) <= 2) and " +
           "(count(../endpoint/pw) <= 1) and " +
           "(count(../endpoint/ac)  <= 1) and " +
           "(count(../endpoint/primary-pw)  <= 1) and " +
           "(count(../endpoint/backup-pw)  <= 1) " {
        description "A VPWS L2VPN instance has at most 2 endpoints " +
                    "and each endpoint has at most 1 pseudowire or " +
                    "1 attachment circuit";
      }
      description "VPWS constraints";
    }
  }

  augment "/ni:network-instances/ni:network-instance/ni:ni-type" +
          "/l2vpn:l2vpn" {
```

```
    when "l2vpn:type = 'l2vpn:vpls-instance-type'" {
      description "Parameters specifically for a VPLS instance";
    }
    description "Augment for parameters for a VPLS instance";
    uses pbb-parameters-grp;
  }

  augment "/ni:network-instances/ni:network-instance/ni:ni-type" +
          "/l2vpn:l2vpn/l2vpn:endpoint" {
    when "../l2vpn:type = 'l2vpn:vpls-instance-type'" {
      description "Endpoint parameter specifically for " +
                  "a VPLS instance";
    }
    description "Augment for endpoint parameters for a VPLS instance";
    leaf split-horizon-group {
      type string;
      description "Identify a split horizon group";
    }
  }

  augment "/ni:network-instances/ni:network-instance/ni:ni-type" +
          "/l2vpn:l2vpn/l2vpn:endpoint" +
          "/l2vpn:ac-or-pw-or-redundancy-grp" +
          "/l2vpn:redundancy-grp/l2vpn:backup" +
          "/l2vpn:backup-pw/l2vpn:backup-pw" {
    when "../../l2vpn:type = 'l2vpn:vpls-instance-type'" {
      description "Backup pseudowire parameter specifically for " +
                  "a VPLS instance";
    }
    description "Augment for backup pseudowire paramters for " +
                "a VPLS instance";
    leaf precedence {
      type uint32;
      description "precedence of the pseudowire";
    }
  }

  /* Notifications */

  notification l2vpn-state-change-notification {
    description "L2VPN and constituents state change notification";
    leaf l2vpn-instance-name {
      type l2vpn-instance-name-ref;
      description "The L2VPN instance name";
    }
    leaf l2vpn-instance-type {
      type leafref {
        path "/ni:network-instances" +
             "/ni:network-instance" +
                  "[ni:name=current()/../l2vpn-instance-name]" +
             "/l2vpn:type";
      }
      description "The L2VPN instance type";
    }
    leaf endpoint {
      type leafref {
        path "/ni:network-instances" +
             "/ni:network-instance" +
                  "[ni:name=current()/../l2vpn-instance-name]" +
             "/l2vpn:endpoint/l2vpn:name";
      }
      description "The endpoint";
    }
    uses endpoint-grp {
      augment "ac-or-pw-or-redundancy-grp/ac" {
        description "Augment for attachment circuit(s) " +
                    "as an endpoint";
        leaf ac {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                    "[ni:name=current()/../l2vpn-instance-name]" +
                 "/l2vpn:endpoint" +
                    "[l2vpn:name=current()/../endpoint]" +
                 "/l2vpn:ac/l2vpn:name";
          }
          description "Related attachment circuit";
        }
      }
      augment "ac-or-pw-or-redundancy-grp/pw" {
        description "Augment for pseudowire(s) as an endpoint";
        leaf pw {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                    "[ni:name=current()/../l2vpn-instance-name]" +
                 "/l2vpn:endpoint[l2vpn:name=current()/../endpoint]" +
                 "/l2vpn:pw/l2vpn:name";
          }
          description "Related pseudowire";
        }
      }
      augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
              "primary/primary-ac" {
        description "Augment for primary-ac";
        leaf primary-ac {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                    "[ni:name=current()/../l2vpn-instance-name]" +
                 "/l2vpn:endpoint" +
                    "[l2vpn:name=current()/../endpoint]" +
                 "/l2vpn:primary-ac/l2vpn:name";
          }
          description "Related primary attachment circuit";
        }
      }
      augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
              "primary/primary-pw" {
        description "Augment for primary-pw";
        leaf primary-pw {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                    "[ni:name=current()/../l2vpn-instance-name]" +
                 "/l2vpn:endpoint" +
                    "[l2vpn:name=current()/../endpoint]" +
                 "/l2vpn:primary-pw/l2vpn:name";
          }
          description "Related primary pseudowire";
        }
      }
      augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
              "backup/backup-ac" {
        description "Augment for backup-ac";
        leaf backup-ac {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                    "[ni:name=current()/../l2vpn-instance-name]" +
                 "/l2vpn:endpoint" +
                    "[l2vpn:name=current()/../endpoint]" +
                 "/l2vpn:backup-ac/l2vpn:name";
          }
          description "Related backup attachment circuit";
        }
      }
      augment "ac-or-pw-or-redundancy-grp/redundancy-grp/" +
              "backup/backup-pw" {
        description "Augment for backup-pw";
        leaf backup-pw {
          type leafref {
            path "/ni:network-instances" +
                 "/ni:network-instance" +
                     "[ni:name=current()/../l2vpn-instance-name]" +
                  "/l2vpn:endpoint" +
                     "[l2vpn:name=current()/../endpoint]" +
                  "/l2vpn:backup-pw/l2vpn:name";
          }
          description "Related backup pseudowire";
        }
      }
    }
    leaf state {
      type identityref {
        base l2vpn-notification-state;
      }
      description "State change notification";
    }
  }
}

<CODE ENDS>
```

Figure 3

5. Security Considerations

The configuration, state, action and notification data defined in this document are 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 means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

The security concerns listed above are, however, no different than faced by other routing protocols. Hence, this draft does not change any underlying security issues inherent in [I-D.ietf-netmod-routing-cfg]

6. IANA Considerations

None.

7. Acknowledgments

The authors would like to acknowledge Giles Heron and others for their useful comments.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[RFC3916] Xiao, X., Ed., McPherson, D., Ed., and P. Pate, Ed., "Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)", RFC 3916, DOI 10.17487/RFC3916, September 2004, <https://www.rfc-editor.org/info/rfc3916>.

[RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <https://www.rfc-editor.org/info/rfc3985>.

[RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, DOI 10.17487/RFC4385, February 2006, <https://www.rfc-editor.org/info/rfc4385>.

[RFC4446] Martini, L., "IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3)", BCP 116, RFC 4446, DOI 10.17487/RFC4446, April 2006, <https://www.rfc-editor.org/info/rfc4446>.

[RFC4448] Martini, L., Ed., Rosen, E., El-Aawar, N., and G. Heron, "Encapsulation Methods for Transport of Ethernet over MPLS Networks", RFC 4448, DOI 10.17487/RFC4448, April 2006, <https://www.rfc-editor.org/info/rfc4448>.

[RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, DOI 10.17487/RFC4664, September 2006, <https://www.rfc-editor.org/info/rfc4664>.

[RFC4665] Augustyn, W., Ed. and Y. Serbest, Ed., "Service Requirements for Layer 2 Provider-Provisioned Virtual Private Networks", RFC 4665, DOI 10.17487/RFC4665, September 2006, <https://www.rfc-editor.org/info/rfc4665>.

[RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <https://www.rfc-editor.org/info/rfc4761>.

[RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <https://www.rfc-editor.org/info/rfc4762>.

[RFC5003] Metz, C., Martini, L., Balus, F., and J. Sugimoto, "Attachment Individual Identifier (AII) Types for Aggregation", RFC 5003, DOI 10.17487/RFC5003, September 2007, <https://www.rfc-editor.org/info/rfc5003>.

[RFC5254] Bitar, N., Ed., Bocci, M., Ed., and L. Martini, Ed., "Requirements for Multi-Segment Pseudowire Emulation Edge-to-Edge (PWE3)", RFC 5254, DOI 10.17487/RFC5254, October 2008, <https://www.rfc-editor.org/info/rfc5254>.

[RFC5659] Bocci, M. and S. Bryant, "An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge", RFC 5659, DOI 10.17487/RFC5659, October 2009, <https://www.rfc-editor.org/info/rfc5659>.

[RFC6073] Martini, L., Metz, C., Nadeau, T., Bocci, M., and M. Aissaoui, "Segmented Pseudowire", RFC 6073, DOI 10.17487/RFC6073, January 2011, <https://www.rfc-editor.org/info/rfc6073>.

[RFC6074] Rosen, E., Davie, B., Radoaca, V., and W. Luo, "Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)", RFC 6074, DOI 10.17487/RFC6074, January 2011, <https://www.rfc-editor.org/info/rfc6074>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6391] Bryant, S., Ed., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, DOI 10.17487/RFC6391, November 2011, <https://www.rfc-editor.org/info/rfc6391>.

[RFC6423] Li, H., Martini, L., He, J., and F. Huang, "Using the Generic Associated Channel Label for Pseudowire in the MPLS Transport Profile (MPLS-TP)", RFC 6423, DOI 10.17487/RFC6423, November 2011, <https://www.rfc-editor.org/info/rfc6423>.

[RFC6478] Martini, L., Swallow, G., Heron, G., and M. Bocci, "Pseudowire Status for Static Pseudowires", RFC 6478, DOI 10.17487/RFC6478, May 2012, <https://www.rfc-editor.org/info/rfc6478>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC6624] Kompella, K., Kothari, B., and R. Cherukuri, "Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling", RFC 6624, DOI 10.17487/RFC6624, May 2012, <https://www.rfc-editor.org/info/rfc6624>.

[RFC7041] Balus, F., Ed., Sajassi, A., Ed., and N. Bitar, Ed., "Extensions to the Virtual Private LAN Service (VPLS) Provider Edge (PE) Model for Provider Backbone Bridging", RFC 7041, DOI 10.17487/RFC7041, November 2013, <https://www.rfc-editor.org/info/rfc7041>.

[RFC7361] Dutta, P., Balus, F., Stokes, O., Calvignac, G., and D. Fedyk, "LDP Extensions for Optimized MAC Address Withdrawal in a Hierarchical Virtual Private LAN Service (H-VPLS)", RFC 7361, DOI 10.17487/RFC7361, September 2014, <https://www.rfc-editor.org/info/rfc7361>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8077] Martini, L., Ed. and G. Heron, Ed., "Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)", STD 84, RFC 8077, DOI 10.17487/RFC8077, February 2017, <https://www.rfc-editor.org/info/rfc8077>.

Appendix A. Example Configuration

This section shows an example configuration using the YANG data model defined in the document.

Appendix B. Contributors

The editors gratefully acknowledge the following people for their contributions to this document.

Reshad Rahman
Cisco Systems, Inc.
Email: rrahman@cisco.com

Kamran Raza
Cisco Systems, Inc.
Email: skraza@cisco.com

Giles Heron
Cisco Systems, Inc.
Email: giheron@cisco.com

Tapraj Singh
Cisco Systems, Inc.
Email: tsingh@cisco.com

Zhenbin Li
Huawei Technologies
Email: lizhenbin@huawei.com

Zhuang Shunwan
Huawei Technologies
Email: Zhuangshunwan@huawei.com

Wang Haibo
Huawei Technologies
Email: rainsword.wang@huawei.com

Sajjad Ahmed
Ericsson
Email: sajjad.ahmed@ericsson.com

Matthew Bocci
Nokia
Email: matthew.bocci@nokia.com

Jorge Rabadan
Nokia

```
                 Email: jorge.rabadan@nokia.com

                 Jonathan Hardwick
                 Metaswitch
                 Email: jonathan.hardwick@metaswitch.com

                 Santosh Esale
                 Juniper Networks
                 Email: sesale@juniper.net

                 Nick Delregno
                 Verizon
                 Email: nick.deregno@verizon.com

                 Luay Jalil
                 Verizon
                 Email: luay.jalil@verizon.com

                 Maria Joecylyn
                 Verizon
                 Email: joecylyn.malit@verizon.com
```

Figure 4

Authors' Addresses

Himanshu Shah
Ciena Corporation

Email: hshah@ciena.com

Patrice Brissette
Cisco Systems, Inc.

Email: pbrisset@cisco.com

Ing-When Chen
The MITRE Corporation

Email: ingwherchen@mitre.org

Iftekar Hussain
Infinera Corporation

Email: ihussain@infinera.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

Kishore Tiruveedhula
Juniper Networks

Email: kishoret@juniper.net