PALS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2016

H. Shah
Ciena Corporation
P. Brissette
R. Rahman
K. Raza
Cisco Systems, Inc.
Z. Li
Z. Shunwan
W. Haibo
Huawei Technologies
I. Chen
Ericsson
M. Bocci
Alcatel-Lucent
J. Hardwick
Metaswitch
S. Esale
Junipr Networks
K. Tiruveedhula
T. Singh
Juniper Networks
I. Hussain
Infinera Corporation
B. Wen
J. Walker
Comcast
N. Delregno
L. Jalil
M. Joecylyn
Verizon

July 06, 2015

YANG Data Model for MPLS-based L2VPN
draft-shah-pals-mpls-l2vpn-yang-00.txt

Abstract

This document describes a YANG data model for Layer 2 VPN services over MPLS networks. These services include Virtual Private Wire Service (VPWS), Virtual Private LAN service (VPLS) and Ethernet Virtual Private Service (EVPN) that uses LDP and BGP signaled Pseudowires. This document mainly focuses on L2VPN VPWS, other services are for future investigations.

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 [RFC6020] 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 introduces a YANG data model for MPLS based Layer 2 VPN services (L2VPN) [RFC4664] as well as switching between the local attachment circuits. The L2VPN services include point-to-point VPWS and Multipoint VPLS and EVPN services. These services are realized by signaling Pseudowires across MPLS networks using LDP [RFC4447][RFC4762] or BGP[RFC4761].

The Yang data model in this document defines Ethernet based Layer 2 services. 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 Ethernet based Layer 2 services will leverage the definitions used in other standards organizations such as IEEE 802.1 and Metro Ethernet Forum (MEF).

The goal is to propose a data object model consisting of building blocks that can be assembled in different order to realize different services. The definition work is undertaken initially by a smaller working group with members representing various vendors and service providers. The VPWS service definitions are covered first followed by VPLS services that build on the data blocks defined for VPWS.

The data model is defined for following constructs that are used for managing the services:

- Configuration
- Operational State
- Executables (Actions)
- Notifications

The document is organized to first define the data model for the configuration, operational state, actions and notifications of VPWS. The L2VPN data object model defined in this document uses the instance centric approach whereby VPWS service attributes are specified for a given VPWS instance.

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

One single top level container, mpls-l2vpn, is defined as a parent for three different second level containers that are vpws-instances, vpls-instances, and common building blocks of AC-templates(Attachment Circuit templates) and pseudowire-templates. This document defines the vpws-instances and templates for AC and Pseudowires. The definition of vpls-instances and evpn-instances is left for future revisions.

The L2VPN services have been defined in the IETF L2VPN working group but leverages the pseudowire technologies that were defined in 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:

- Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3) [RFC3916]
- Pseudo-wire Emulation Edge-to-Edge (PWE3) Architecture [RFC3985]
- IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3) [RFC4446]
- Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP) [RFC4447]
- Encapsulation Methods for Transport of Ethernet over MPLS Networks [RFC4448]

- Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN [RFC4385]
- Requirements for Multi-Segment Pseudowire Emulation Edge-to-Edge (PWE3) [RFC5254]
- An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge [RFC5659]
- Segmented Pseudowire [RFC6073]
- Framework for Layer 2 Virtual Private Networks [RFC4664]
- Service Requirements for Layer 2 Provider-Provisioned Virtual Private Networks [RFC4665]
- Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling [RFC4761]
- Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling [RFC4762]
- Attachment Individual Identifier (AII) Types for Aggregation [RFC5003]
- Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs) [RFC6074]
- Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network [RFC6391]
- Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling [RFC6624]
- Extensions to the Virtual Private LAN Service (VPLS) Provider Edge (PE) Model for Provider Backbone Bridging [RFC7041]
- LDP Extensions for Optimized MAC Address Withdrawal in a Hierarchical Virtual Private LAN Service (H-VPLS) [RFC7361]
- Using the generic associated channel label for Pseudowire in the MPLS Transport Profile [RFC6423]
- Pseudowire status for static pseudowire [RFC6478]

Note that while pseudowire over MPLS-TP related work is in scope, the initial effort will only address definitions of object model for VPWS services 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.

```
template-ref AC // AC
                   template
                   attributes

template-ref PW // PW
                   template
                   attributes

vpws-instance name // container

        svc-type

        // list of AC and PW being used
        AC-1 // container
                template-ref AC
                attribute-override
        PW-2 // container
                template-ref PW
                attribute-override
        PW-3 // container
                template-ref PW
                attribute-override

        // ONLY 2 endpoints!!!
        endpoint-A // container
                        AC-1 // reference

        endpoint-Z // container
                redundancy-grp // container
                        PW-2 // reference
                        PW-3 // reference
```

Figure 1

3.2. L2VPN Common

3.2.1. ac-templates

The ac-templates container contains a list of ac-template. Each ac-template defines a list of AC attributes that are part of native services but associated and processed within the context of L2VPN. For instance, Ethernet VLAN tag imposition, disposition and translation or CVID-bundling would be part of this template.

3.2.2. 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.

3.3. VPWS

3.3.1. ac list

Each VPWS instance defines a list of AC which are cross-connected by the service. Each entry of the AC consists of one ac-template with predefined attributes and values, but also defines attributes that override the attributes defined in referenced ac-template.

3.3.2. pw list

Each VPWS instance defines a list of PW which are cross-connected by the service. Each entry of the PW consists of one pw-template with pre-defined attributes and values, but also defines attributes that override those defined in referenced pw-template. No restrictions are placed on type of signaling (i.e. LDP or BGP) used for a given PW. It is entirely possible to define two PWs, one signaled by LDP and other by BGP.

3.3.3. 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 -

- primary and backup AC
- primary and backup PW
- primary AC and backup PW
- primary PW and backup AC

3.3.4. endpoint container

The endpoint container holds 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.

3.3.5. vpws-instances container

The vpws-instances container contains a list of vpws-instance. Each entry of the vpws-instance represents a layer-2 cross-connection of two endpoints. This model defines three possible types of endpoints, ac, pw, and redundancy-grp, and allows a vpws-instance to cross-connect any one type of endpoint to all other types of endpoint.

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

```
module: ietf-mpls-l2vpn
+--rw mpls-l2vpn
   +--rw common
   |  +--rw pw-templates
   |  |  +--rw pw-template* [name]
   |  |     +--rw name              string
   |  |     +--rw mtu?              uint32
   |  |     +--rw cw-negotiation?   cw-negotiation-type
   |  |     +--rw tunnel-policy?    string
   |  +--rw ac-templates
   |     +--rw ac-template* [name]
   |        +--rw name    string
   +--rw vpws-instances
   |  +--rw vpws-instance* [instance-name]
   |     +--rw instance-name     string
   |     +--rw description?      string
   |     +--rw service-type?     l2vpn-service-type
   |     +--rw discovery-type?   l2vpn-discovery-type
   |     +--rw signaling-type    l2vpn-signaling-type
   |     +--rw bgp-parameters
   |     |  +--rw common
   |     |  |  +--rw route-distinguisher?   string
   |     |  |  +--rw vpn-targets* [rt-value]
   |     |  |     +--rw rt-value    string
   |     |  |     +--rw rt-type     bgp-rt-type
   |     |  +--rw discovery
   |     |  |  +--rw vpn-id?   string
```

```
|     |  +--rw signaling
|     |     +--rw site-id?      uint16
|     |     +--rw site-range?   uint16
|     +--rw pw* [name]
|     |  +--rw name              string
|     |  +--rw cw-negotiation?   cw-negotiation-type
|     |  +--rw template?         pw-template-ref
|     |  +--rw vccv-ability?     boolean
|     |  +--rw tunnel-policy?    string
|     |  +--rw request-vlanid?   uint16
|     |  +--rw vlan-tpid?        string
|     |  +--rw ttl?              uint8
|     |  +--rw (pw-type)?
|     |     +--:(ldp-pw)
|     |     |  +--rw peer-ip?          inet:ip-address
|     |     |  +--rw pw-id?            uint32
|     |     |  +--rw transmit-label?   uint32
|     |     |  +--rw receive-label?    uint32
|     |     |  +--rw icb?              boolean
|     |     +--:(bgp-pw)
|     |     |  +--rw remote-pe-id?     inet:ip-address
|     |     +--:(bgp-ad-pw)
|     |        +--rw remote-ve-id?     uint16
|     +--rw ac* [name]
|     |  +--rw name                       string
|     |  +--rw template?                  ac-template-ref
|     |  +--rw pipe-mode?                 enumeration
|     |  +--rw link-discovery-protocol?  link-discovery-protocol-type
|     +--rw endpoint-a
|     |  +--rw (ac-or-pw-or-redundancy-grp)?
|     |     +--:(ac)
|     |     |  +--rw ac?                -> ../../ac/name
|     |     +--:(pw)
|     |     |  +--rw pw?                -> ../../pw/name
|     |     +--:(redundancy-grp)
|     |     |  +--rw (primary)
|     |     |  |  +--:(primary-pw)
|     |     |  |  |  +--rw primary-pw?        -> ../../pw/name
|     |     |  |  +--:(primary-ac)
|     |     |  |     +--rw primary-ac?        -> ../../ac/name
|     |     |  +--rw (backup)
|     |     |  |  +--:(backup-pw)
|     |     |  |  |  +--rw backup-pw?         -> ../../pw/name
|     |     |  |  +--:(backup-ac)
|     |     |  |     +--rw backup-ac?         -> ../../ac/name
|     |     |  +--rw protection-mode?   enumeration
|     |     +--:(reroute-mode)
|     |     |  +--rw reroute-mode?      enumeration
```

```
|     |     +--:(reroute-delay)
|     |     |  +--rw reroute-delay?     uint16
|     |     +--:(dual-receive)
|     |     |  +--rw dual-receive?      boolean
|     |     +--:(revert)
|     |     |  +--rw revert?            boolean
|     |     +--:(revert-delay)
|     |        +--rw revert-delay?      uint16
|     +--rw endpoint-z
|        +--rw (ac-or-pw-or-redundancy-grp)?
|           +--:(ac)
|           |  +--rw ac?                -> ../../ac/name
|           +--:(pw)
|           |  +--rw pw?                -> ../../pw/name
|           +--:(redundancy-grp)
|           |  +--rw (primary)
|           |  |  +--:(primary-pw)
|           |  |  |  +--rw primary-pw?        -> ../../pw/name
|           |  |  +--:(primary-ac)
|           |  |     +--rw primary-ac?        -> ../../ac/name
|           |  +--rw (backup)
|           |  |  +--:(backup-pw)
|           |  |  |  +--rw backup-pw?         -> ../../pw/name
|           |  |  +--:(backup-ac)
|           |  |     +--rw backup-ac?         -> ../../ac/name
|           |  +--rw protection-mode?   enumeration
|           +--:(reroute-mode)
|           |  +--rw reroute-mode?      enumeration
|           +--:(reroute-delay)
|           |  +--rw reroute-delay?     uint16
|           +--:(dual-receive)
|           |  +--rw dual-receive?      boolean
|           +--:(revert)
|           |  +--rw revert?            boolean
|           +--:(revert-delay)
|              +--rw revert-delay?      uint16
+--rw vpls-instances
```

Figure 2

4. YANG Module

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

```
<CODE BEGINS> file "ietf-mpls-l2vpn@2015-06-30.yang"
module ietf-mpls-l2vpn {
  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-l2vpn";
  prefix "mpls-l2vpn";

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

  organization  "ietf";
  contact       "ietf";
  description   "mpls-l2vpn";
  revision "2015-06-30" {
    description "Initial revision";
    reference   "";
  }

  /* identities */

  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";
  }
```

```
     /* typedefs */

     typedef l2vpn-service-type {
       type enumeration {
         enum ethernet {
           description "Ethernet service";
         }
         enum ATM {
           description "Asynchronous Transfer Mode";
         }
         enum FR {
           description "Frame-Relay";
         }
         enum TDM {
           description "Time Division Multiplexing";
         }
       }
       description "L2VPN service type";
     }

     typedef l2vpn-discovery-type {
       type enumeration {
         enum manual {
           description "Manual configuration";
         }
         enum bgp-ad {
           description "Border Gateway Protocol (BGP) auto-discovery";
         }
       }
       description "L2VPN discovery type";
     }

     typedef l2vpn-signaling-type {
       type enumeration {
         enum static {
           description "Static configuration of labels (no signaling)";
         }
         enum ldp {
           description "Label Distribution Protocol (LDP) signaling";
         }
         enum bgp {
           description "Border Gateway Protocol (BGP) signaling";
         }
       }
       description "L2VPN signaling type";
     }

     typedef bgp-rt-type {
```

```
      type enumeration {
        enum import {
          description "For import";
        }
        enum export {
          description "For export";
        }
        enum both {
          description "For both import and export";
        }
      }
      description "BGP route-target type. Import from BGP YANG";
    }

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

    typedef pw-template-ref {
      type leafref {
        path "/l2vpn/common/pw-templates/pw-template/name";
      }
      description "pw-template-ref";
    }

    typedef ac-template-ref {
      type leafref {
        path "/l2vpn/common/ac-templates/ac-template/name";
      }
      description "ac-tempalte-ref";
    }
```

```
     /* groupings */

     grouping vpws-endpoint {
       description
         "A vpws-endpoing could either be an ac or a pw";
       choice ac-or-pw-or-redundancy-grp {
         description "A choice ofattachment circuit or " +
                     "pseudowire or redundancy group";
         case ac {
           leaf ac {
             type leafref {
               path "../../ac/name";
             }
             description "reference to an attachment circuit";
           }
         }
         case pw {
           leaf pw {
             type leafref {
               path "../../pw/name";
             }
             description "reference to a pseudowire";
           }
         }
         case redundancy-grp {
           choice primary {
             mandatory true;
             description "primary options";
             case primary-pw  {
               leaf primary-pw {
                 type leafref {
                   path "../../pw/name";
                 }
                 description "primary pseudowire";
               }
             }
             case primary-ac {
               leaf primary-ac {
                 type leafref {
                   path "../../ac/name";
                 }
                 description "primary attachment circuit";
               }
             }
           }
           choice backup {
             mandatory true;
             description "backup options";
```

```
            case backup-pw {
              leaf backup-pw {
                type leafref {
                  path "../../pw/name";
                }
                description "backup pseudowire";
              }
            }
            case backup-ac {
              leaf backup-ac {
                type leafref {
                  path "../../ac/name";
                }
                description "backup attachment circuit";
              }
            }
          }
          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 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 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";
          /* This is called "revertive" during the discussion. */
        }
        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";
          /* This is called "wtr" during discussion. */
        }
      }
    }

    /* We can define vpls-endpoing-grp that has the same structure as
     * vpws-endpoing-grp, but has more endpoint options.
     */

    /* L2VPN YANG Model */

    container l2vpn {
      description "l2vpn";
```

```
      container common {
        description "common l2pn attributes";
        container pw-templates {
          description "pw-templates";
          list pw-template {
            key "name";
            description "pw-template";
            leaf name {
              type string;
              description "name";
            }
            leaf mtu {
              type uint32;
              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";
            }
          }
        }
        container ac-templates {
          description "attachment circuit templates";
          /* To be fleshed out in future revisions */
          list ac-template {
            key "name";
            description "ac-template";
            leaf name {
              type string;
              description "name";
            }
          }
        }
      }
      container vpws-instances {
        description "vpws-instances";
        list vpws-instance {
          key "instance-name";
          description "A VPWS instance";
          leaf instance-name {
            type string;
            description "Name of VPWS instance";
```

```
          }
          leaf description {
            type string;
            description "Description of the VPWS instance";
          }
          leaf service-type {
            type l2vpn-service-type;
            default ethernet;
            description "VPWS service type";
          }
          leaf discovery-type {
            type l2vpn-discovery-type;
            default manual;
            description "VPWS discovery type";
          }
          leaf signaling-type {
            type l2vpn-signaling-type;
            mandatory true;
            description "VPWS signaling type";
          }
          container bgp-parameters {
            description "Parameters for BGP";
            container common {
              when "../../discovery-type = 'bgp-ad'" {
                description "Check discovery type: " +
                            "Can only configure BGP discovery if " +
                            "discovery type is BGP-AD";
              }
              description "Common BGP parameters";
              leaf route-distinguisher {
                type string;
                description "BGP RD";
              }
              list vpn-targets {
                key rt-value;
                description "Route Targets";
                leaf rt-value {
                  type string;
                  description "Route-Target value";
                }
                leaf rt-type {
                  type bgp-rt-type;
                  mandatory true;
                  description "Type of RT";
                }
              }
            }
            container discovery {
```

```
            when "../../discovery-type = 'bgp-ad'" {
              description "BGP parameters for discovery: " +
                          "Can only configure BGP discovery if " +
                          "discovery type is BGP-AD";
            }
            description "BGP parameters for discovery";
            leaf vpn-id {
              type string;
              description "VPN ID";
            }
          }
          container signaling {
            when "../../signaling-type = 'bgp'" {
              description "Check signaling type: " +
                          "Can only configure BGP signaling if " +
                          "signaling type is BGP";
            }
            description "BGP parameters for signaling";
            leaf site-id {
              type uint16;
              description "Site ID";
            }
            leaf site-range {
              type uint16;
              description "Site Range";
            }
          }
        }
        list pw {
          key "name";
          description "pseudowire";
          leaf name {
            type string;
            description "pseudowire name";
          }
          leaf cw-negotiation {
            type cw-negotiation-type;
            default "preferred";
            description "Override the control-word negotiation " +
                        "preference specified in the " +
                        "pseudowire template.";
          }
          leaf template {
            type pw-template-ref;
            description "pseudowire template";
          }
          leaf vccv-ability {
            type boolean;
```

```
            description "vccvability";
          }
          leaf tunnel-policy {
            type string;
            description "Used to override the tunnel policy name " +
                        "specified in the pseduowire template";
          }
          leaf request-vlanid {
            type uint16;
            description "request vlanid";
          }
          leaf vlan-tpid {
            type string;
            description "vlan tpid";
          }
          leaf ttl {
            type uint8;
            description "time-to-live";
          }
          choice pw-type {
            description "A choice of pseudowire type";
            case ldp-pw {
              leaf peer-ip {
                type inet:ip-address;
                description "peer IP address";
              }
              leaf pw-id {
                type uint32;
                description "pseudowire id";
              }
              leaf transmit-label {
                type uint32;
                description "transmit lable";
              }
              leaf receive-label {
                type uint32;
                description "receive label";
              }
              leaf icb {
                type boolean;
                description "inter-chassis backup";
              }
            }
            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";
              }
            }
          }
        }
        list ac {
          key "name";
          description "attachment circuit";
          leaf name {
            type string;
            description "name";
          }
          leaf template {
            type ac-template-ref;
            description "attachment circuit template";
          }
          leaf pipe-mode {
            type enumeration {
              enum "pipe" {
                value 0;
                description "regular pipe mode";
              }
              enum "short-pipe" {
                value 1;
                description "short pipe mode";
              }
              enum "uniform" {
                value 2;
                description "uniform pipe mode";
              }
            }
            description "pipe mode";
          }
          leaf link-discovery-protocol {
            type link-discovery-protocol-type;
            description "link discovery protocol";
          }
        }
        container endpoint-a {
          description "endpoint-a";
          uses vpws-endpoint;
        }
        container endpoint-z {
          description "endpoint-z";
```

```
            uses vpws-endpoint;
          }
        }
      }
      container vpls-instances {
        /* To be fleshed out in future revisions */
        description "vpls-instances";
      }
    }
  }

  <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 TBD 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, March 1997.

8.2. Informative References

[RFC3916] Xiao, X., McPherson, D., and P. Pate, "Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)", RFC 3916, September 2004.

[RFC3985] Bryant, S. and P. Pate, "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, March 2005.

[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, February 2006.

[RFC4446] Martini, L., "IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3)", BCP 116, RFC 4446, April 2006.

[RFC4447] Martini, L., Rosen, E., El-Aawar, N., Smith, T., and G. Heron, "Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)", RFC 4447, April 2006.

[RFC4448] Martini, L., Rosen, E., El-Aawar, N., and G. Heron, "Encapsulation Methods for Transport of Ethernet over MPLS Networks", RFC 4448, April 2006.

[RFC4664] Andersson, L. and E. Rosen, "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, September 2006.

[RFC4665] Augustyn, W. and Y. Serbest, "Service Requirements for Layer 2 Provider-Provisioned Virtual Private Networks", RFC 4665, September 2006.

[RFC4761] Kompella, K. and Y. Rekhter, "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, January 2007.

[RFC4762] Lasserre, M. and V. Kompella, "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, January 2007.

[RFC5003] Metz, C., Martini, L., Balus, F., and J. Sugimoto, "Attachment Individual Identifier (AII) Types for Aggregation", RFC 5003, September 2007.

[RFC5254] Bitar, N., Bocci, M., and L. Martini, "Requirements for Multi-Segment Pseudowire Emulation Edge-to-Edge (PWE3)", RFC 5254, October 2008.

[RFC5659] Bocci, M. and S. Bryant, "An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge", RFC 5659, October 2009.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6073] Martini, L., Metz, C., Nadeau, T., Bocci, M., and M. Aissaoui, "Segmented Pseudowire", RFC 6073, January 2011.

[RFC6074] Rosen, E., Davie, B., Radoaca, V., and W. Luo, "Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)", RFC 6074, January 2011.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6391] Bryant, S., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, November 2011.

[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, November 2011.

[RFC6478] Martini, L., Swallow, G., Heron, G., and M. Bocci, "Pseudowire Status for Static Pseudowires", RFC 6478, May 2012.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC6624] Kompella, K., Kothari, B., and R. Cherukuri, "Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling", RFC 6624, May 2012.

[RFC7041] Balus, F., Sajassi, A., and N. Bitar, "Extensions to the Virtual Private LAN Service (VPLS) Provider Edge (PE) Model for Provider Backbone Bridging", RFC 7041, November 2013.

[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, September 2014.

Authors' Addresses

Himanshu Shah
Ciena Corporation

Email: hshah@ciena.com

Patrice Brissette
Cisco Systems, Inc.

Email: pbrisset@cisco.com

Reshad Rahman
Cisco Systems, Inc.

Email: rrahman@cisco.com

Kamran Raza
Cisco Systems, Inc.

Email: skraza@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

Ing-When Chen
Ericsson

Email: ing-wher.chen@ericsson.com

Mathew Bocci
Alcatel-Lucent

Email: mathew.bocci@alcatel-lucent.com

Jonathan Hardwick
Metaswitch

Email: jonathan.hardwick@metaswitch.com

Santosh Esale
Junipr Networks

Email: sesale@juniper.net

Kishore Tiruveedhula
Juniper Networks

Email: kishoret@juniper.net

Tapraj Singh
Juniper Networks

Email: tsingh@juniper.net

Iftekar Hussain
Infinera Corporation

Email: ihussain@infinera.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

Jason Walker
Comcast

Email: jason_walker2@cable.comcast.com

Nick Delregno
Verizon

Email: nick.deregno@verizon.com

Luay Jalil
Verizon

Email: luay.jalil@verizon.com

Maria Joecylyn
Verizon

Email: joecylyn.malit@verizon.com