L2SM Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 13, 2018

B. Wen
Comcast
G. Fioccola, Ed.
Telecom Italia
C. Xie
China Telecom
L. Jalil
Verizon

April 11, 2018

A YANG Data Model for L2VPN Service Delivery
draft-ietf-l2sm-l2vpn-service-model-10

Abstract

This document defines a YANG data model that can be used to configure a Layer 2 Provider Provisioned VPN service. It is up to a management system to take this as an input and generate specific configurations models to configure the different network elements to deliver the service. How configuration of network elements is done is out of scope of the document.

The YANG model in this document includes support for point-to-point Virtual Private Wire Services (VPWS) and multipoint Virtual Private LAN services (VPLS) that use Pseudowires signaled using the Label Distribution Protocol (LDP) and the Border Gateway Protocol (BGP) as described in RFC4761 and RFC6624.

The YANG model in this document conforms to the Network Management Datastore Architecture defined in RFC8342.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 13, 2018.

Table of Contents

1. Introduction

This document defines a YANG data model for Layer 2 VPN (L2VPN) service. This model defines service configuration elements that can be used in communication protocols between customers and network

operators. Those elements can also be used as input to automated control and configuration applications and and generate specific configurations models to configure the different network elements to deliver the service. How configuration of network elements is done is out of scope of the document.

Further discussion of the way that services are modeled in YANG and the relationship between "customer service models" like the one described in this document and configuration models can be found in [RFC8309] and [RFC8199]. Section 4 and Section 6 also provide more information of how this service model could be used and how it fits into the overall modeling architecture.

The YANG model in this document includes support for point-to-point Virtual Private Wire Services (VPWS) and multipoint Virtual Private LAN services (VPLS) that use Pseudowires signaled using the Label Distribution Protocol (LDP) and the Border Gateway Protocol (BGP) as described in [RFC4761] and [RFC6624]. It also conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

1.1. Terminology

The following terms are defined in [RFC6241] and are not redefined here:

- client
- configuration data
- server
- state data

The following terms are defined in [RFC7950] and are not redefined here:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC7950].

1.1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Tree diagram

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Definitions

This document uses the following terms:

Service Provider (SP): The organization (usually a commercial undertaking) responsible for operating the network that offers VPN services to clients and customers.

Customer Edge (CE) Device: Equipment that is dedicated to a particular customer and is directly connected to one or more PE devices via attachment circuits. A CE is usually located at the customer premises, and is usually dedicated to a single VPN, although it may support multiple VPNs if each one has separate attachment circuits. The CE devices can be routers, bridges, switches, or hosts.

Provider Edge (PE) Device: Equipment managed by the SP that can support multiple VPNs for different customers, and is directly connected to one or more CE devices via attachment circuits. A PE is usually located at an SP point of presence (PoP) and is managed by the SP.

Virtual Private LAN Service (VPLS): A VPLS is a provider service that emulates the full functionality of a traditional Local Area Network (LAN). A VPLS makes it possible to interconnect several LAN segments over a packet switched network (PSN) and makes the remote LAN segments behave as one single LAN.

Virtual Private Wire Service (VPWS): A VPWS is a point-to-point circuit (i.e., link) connecting two CE devices. The link is established as a logical Layer 2 circuit through a packet switched network. The CE in the customer network is connected to a PE in the provider network via an Attachment Circuit (AC): the AC is either a physical or a logical circuit. A VPWS differs from a VPLS in that the VPLS is point-to-multipoint, while the VPWS is

point-to-point. In some implementations, a set of VPWSs is used to create a multi-site L2VPN network.

Pseudowire(PW): A pseudowire is an emulation of a native service over a packet switched network (PSN). The native service may be ATM, frame relay, Ethernet, low-rate TDM, or SONET/SDH, while the PSN may be MPLS, IP (either IPv4 or IPv6), or L2TPv3.

MAC-VRF: A Virtual Routing and Forwarding table for Media Access Control (MAC) addresses on a PE. It is sometime also referred to as a Virtual Switching Instance (VSI).

UNI: User to Network Interface. The physical demarcation point between the responsibility of Customer and the responsibility of Provider.

NNI: Network to Network Interface. A reference point representing the boundary between two Networks that are operated as separate administrative domains. The two networks may belong to the same provider or to two different providers.

This document uses the following abbreviations:

BSS: Business Support System

BUM: Broadcast-UnknownUnicast-Multicast

CoS: Class of Service

LAG: Link Aggregation Group

LLDP: Link Level Discovery Protocol

OAM: Operations, Administration, and Maintenance

OSS: Operations Support System

PDU: Protocol Data Unit

QoS: Quality of Service

3. The Layer 2 VPN Service Model

A Layer 2 VPN (L2VPN) service is a collection of sites that are authorized to exchange traffic between each other over a shared infrastructure of a common technology. This L2VPN service model (L2SM) provides a common understanding of how the corresponding L2VPN service is to be deployed over the shared infrastructure.

This document presents the L2SM using the YANG data modeling language [RFC7950] as a formal language that is both human-readable and parsable by software for use with protocols such as NETCONF [RFC6241] and RESTCONF [RFC8040].

This service model is limited to VPWS and VPLS based VPNs as described in [RFC4761] and [RFC6624], EVPN as described in [RFC7432].

3.1. Layer 2 VPN Service Types

From a technology perspective, a set of basic L2VPN service types include:

- Point-to-point Virtual Private Wire Services (VPWSs) that use LDP-signaled Pseudowires or L2TP-signaled Pseudowires [RFC6074];
- Multipoint Virtual Private LAN Services (VPLSs) that use LDP-signaled Pseudowires or L2TP-signaled Pseudowires [RFC6074];
- Multipoint Virtual Private LAN Services (VPLSs) that use a Border Gateway Protocol (BGP) control plane as described in [RFC4761] and [RFC6624];
- IP-Only LAN-Like Services (IPLSs) that are a functional subset of VPLS services [RFC7436];
- BGP MPLS-based Ethernet VPN Services as described in [RFC7432] and [RFC7209];
- Ethernet VPN VPWS specified in [RFC8214] and [RFC7432];

3.2. Layer 2 VPN Physical Network Topology

Figure 1 depicts a typical service provider’s physical network topology. Most service providers have deployed an IP, MPLS, or Segment Routing (SR) multi-service core infrastructure. Ingress Layer 2 service frames will be mapped to either an Ethernet Pseudowire (PWE) or a VXLAN PE-to-PE tunnel. The details of these tunneling mechanism are at the provider’s discretion and not part of the L2SM.

An L2VPN provides end-to-end L2 connectivity over this multi-service core infrastructure between two or more customer locations or a collection of sites. Attachment Circuits are placed between CE devices and PE Devices that backhaul layer 2 service frames from the customer over the access network to the Provider Network or remote Site. The demarcation point (i.e., UNI) between the customer and service provider can be either placed between Customer nodes and the

Customer Edge Device, or between the Customer Edge Device and the Provider Edge Device. The actual bearer connection between CE and PE will be described in the L2SM model.

The service provider may also choose a Seamless MPLS approach to expand the PWE or VXLAN tunnel between sites.

The service provider may leverage multi-protocol BGP to auto-discover and signal the PWE or VXLAN tunnel end points.

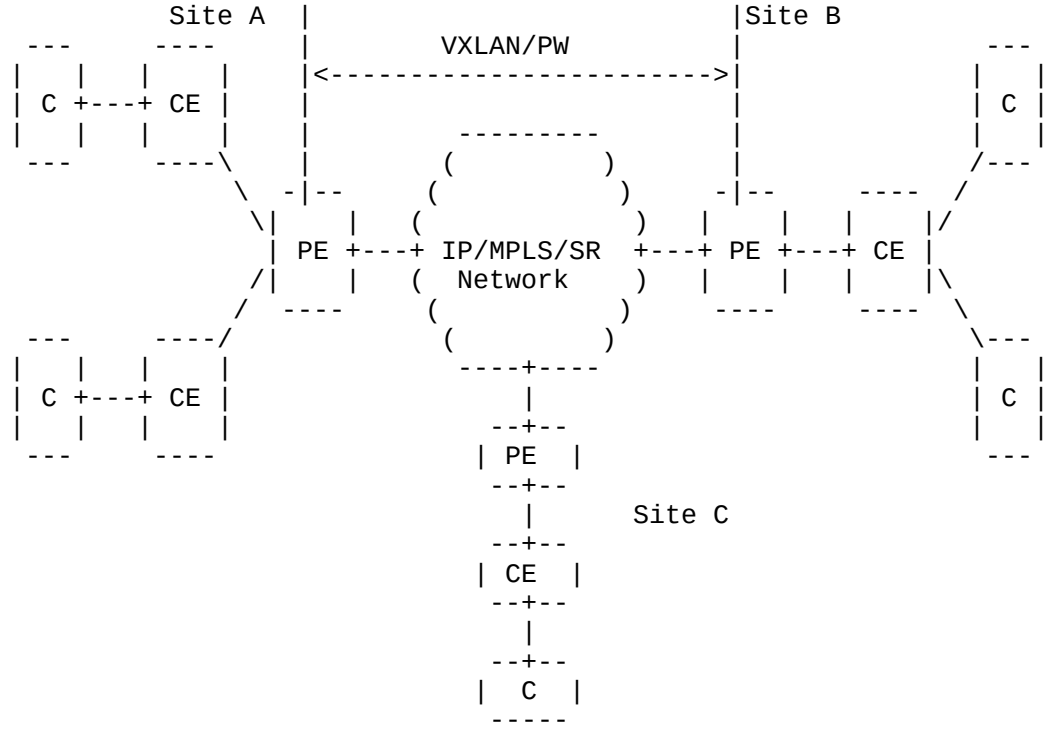


Figure 1: Reference Network for the Use of the L2VPN Service Model

From the customer perspective, however, all the customer edge devices are connected over a simulated LAN environment as shown in Figure 2. Broadcast and multicast packets are sent to all participants in the same bridge domain.

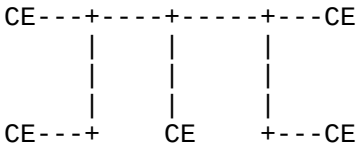


Figure 2: Customer View of the L2VPN

4. Service Data Model Usage

The L2VPN service model provides an abstracted interface to request, configure, and manage the components of an L2VPN service. The model is used by a customer who purchases connectivity and other services from an SP to communicate with that SP.

A typical usage for this model is as an input to an orchestration layer that is responsible for translating it into configuration commands for the network elements that deliver/enable the service. The network elements may be routers, but also servers (like AAA) that are necessary within the network.

The configuration of network elements may be done using the Command Line Interface (CLI), or any other configuration (or "southbound") interface such as NETCONF [RFC6241] in combination with device-specific and protocol-specific YANG models.

This way of using the service model is illustrated in Figure 3 and described in more detail in [RFC8309] and [RFC8199]. The split of the orchestration function between a "Service Orchestrator" and a "Network Orchestrator" is clarified in [RFC8309]. The usage of this service model is not limited to this example: it can be used by any component of the management system, but not directly by network elements.

The usage and structure of this model should be compared to the Layer 3 VPN service model defined in [RFC8299].

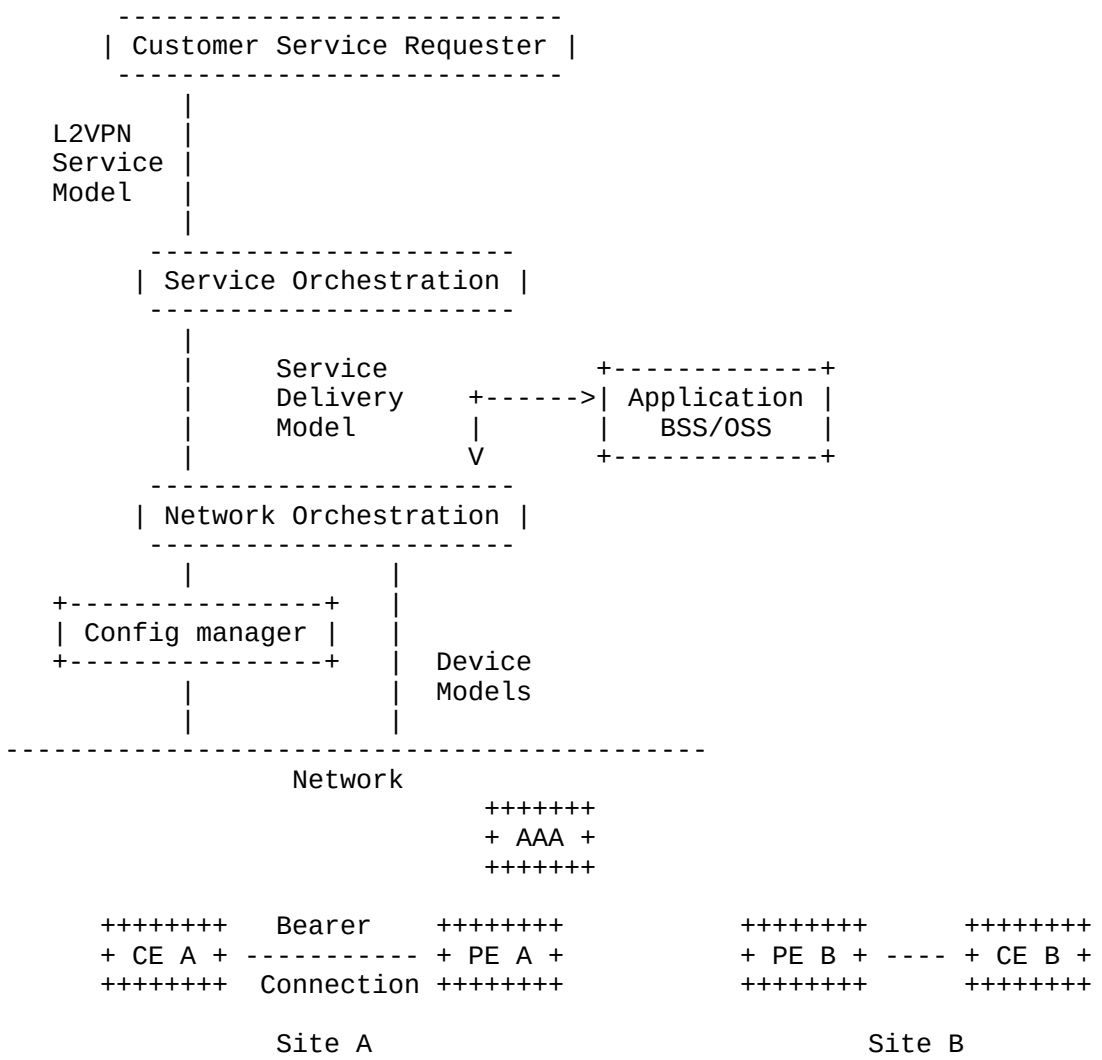


Figure 3: Reference Architecture for the Use of the L2VPN Service Model

The MEF Forum (MEF) has also developed an architecture for network management and operation, but the work of the MEF embraces all aspects of Lifecycle Service Orchestration, including billing, SLAs, order management, and lifecycle management. The IETF's work on service models is typically smaller and offers a simple, self-contained service YANG module. See more details in [RFC8309].

5. Design of the Data Model

The L2SM model is structured in a way that allows the provider to list multiple circuits of various service types for the same customer. A circuit represents an end-to-end connection between two or more locations of Customers.

The YANG module is divided into two main containers: vpn-services and sites. The vpn-svc container under vpn-services defines global parameters for the VPN service for a specific customer.

A site contains at least one network access (i.e., site network accesses providing access to the sites defined in Section 5.3.2) and there may be multiple network accesses in case of multihoming. The site to network access attachment is done through a bearer with a Layer 2 connection on top. The bearer refers to properties of the attachment that are below layer 2 while the connection refers to layer 2 protocol oriented properties. The bearer may be allocated dynamically by the service provider and the customer may provide some constraints or parameters to drive the placement.

Authorization of traffic exchange is done through what we call a VPN policy or VPN topology defining routing exchange rules between sites.

An end to end Multi-segment connectivity can be realized using a combination of per site connectivity and per segment connectivity at different segments.

The figure below describe the overall structure of the YANG module:

```
module: ietf-l2vpn-svc
+--rw l2vpn-svc
  +--rw vpn-profiles
  |  +--rw valid-provider-identifiers
  |     +--rw cloud-identifier*  string{cloud-access}?
  |     +--rw qos-profile-identifier* string
  |     +--rw bfd-profile-identifier* string
  |     +--rw remote-carrier-identifier* string
  +--rw vpn-services
  |  +--rw vpn-service* [vpn-id]
  |     +--rw vpn-id                      svc-id
  |     +--rw vpn-svc-type?               identityref
  |     +--rw customer-name?              string
  |     +--rw svc-topo?                   identityref
  |     +--rw cloud-accesses {cloud-access}?
  |     |  +--rw cloud-access* [cloud-identifier]
  |     |     +--rw cloud-identifier
  |     |     |    -> /l2vpn-svc/vpn-profiles/
```

```
|     |     |      valid-provider-identifiers/cloud-identifier
|     |     +--rw (list-flavor)?
|     |        +--:(permit-any)
|     |        |  +--rw permit-any?         empty
|     |        +--:(deny-any-except)
|     |        |  +--rw permit-site*
|     |        |  :    -> /l2vpn-svc/sites/site/site-id
|     |        +--:(permit-any-except)
|     |           +--rw deny-site*
|     |                -> /l2vpn-svc/sites/site/site-id
|     +--rw frame-delivery {frame-delivery}?
|     |  +--rw customer-tree-flavors
|     |  |  +--rw tree-flavor*   identityref
|     |  +--rw bum-frame-delivery
|     |  |  +--rw bum-frame-delivery* [frame-type]
|     |  |     +--rw frame-type       identityref
|     |  |     +--rw delivery-mode?   identityref
|     |  +--rw multicast-gp-port-mapping    identityref
|     +--rw extranet-vpns {extranet-vpn}?
|     |  +--rw extranet-vpn* [vpn-id]
|     |     +--rw vpn-id              svc-id
|     |     +--rw local-sites-role?   identityref
|     +--rw ce-vlan-preservation        boolean
|     +--rw ce-vlan-cos-preservation    boolean
|     +--rw carrierscarrier?            boolean {carrierscarrier}?
+--rw sites
  +--rw site* [site-id]
   +--rw site-id                                string
   +--rw site-vpn-flavor?                       identityref
   +--rw devices
   |  +--rw device* [device-id]
   |     +--rw device-id     string
   |     +--rw location
   |     |    -> ../../../locations/location/location-id
   |     +--rw management
   |        +--rw transport?   identityref
   |        +--rw address?     inet:ip-address
   +--rw management
   |  +--rw type    identityref
   +--rw locations
   |  +--rw location* [location-id]
   |     +--rw location-id     string
   |     +--rw address?        string
   |     +--rw postal-code?       string
   |     +--rw state?          string
   |     +--rw city?           string
   |     +--rw country-code?   string
   +--rw site-diversity {site-diversity}?
```

```
|  +--rw groups
|     +--rw group* [group-id]
|        +--rw group-id    string
+--rw vpn-policies
|  +--rw vpn-policy* [vpn-policy-id]
|     +--rw vpn-policy-id    string
|     +--rw entries* [id]
|        +--rw id         string
|        +--rw filters
|        |  +--rw filter* [type]
|        |     +--rw type       identityref
|        |     +--rw lan-tag*   uint32 {lan-tag}?
|        +--rw vpn* [vpn-id]
|           +--rw vpn-id
|           |    -> /l2vpn-svc/vpn-services/
|           |            vpn-service/vpn-id
|           +--rw site-role?   identityref
+--rw service
|  +--rw qos {qos}?
|  |  +--rw classification-policy
|  |  |  +--rw rule* [id]
|  |  |     +--rw id                    string
|  |  |     +--rw (match-type)?
|  |  |     |  +--:(match-flow)
|  |  |     |  |  +--rw match-flow
|  |  |     |  |     +--rw dscp?           inet:dscp
|  |  |     |  |     +--rw dot1q?          uint16
|  |  |     |  |     +--rw pcp?            uint8
|  |  |     |  |     +--rw src-mac?        yang:mac-address
|  |  |     |  |     +--rw dst-mac?        yang:mac-address
|  |  |     |  |     +--rw color-type?     identityref
|  |  |     |  |     +--rw target-sites*
|  |  |     |  |     |               svc-id {target-sites}?
|  |  |     |  |     +--rw any?            empty
|  |  |     |  |     +--rw vpn-id?         svc-id
|  |  |     |  +--:(match-application)
|  |  |     |     +--rw match-application?   identityref
|  |  |     +--rw target-class-id?     string
|  |  +--rw qos-profile
|  |     +--rw (qos-profile)?
|  |        +--:(standard)
|  |        |  +--rw profile?
|  |        |       -> /l2vpn-svc/vpn-profiles/
|  |        |              valid-provider-identifiers/
|  |        |              qos-profile-identifier
|  |        +--:(custom)
|  |           +--rw classes {qos-custom}?
|  |              +--rw class* [class-id]
```

```
|  |                +--rw class-id        string
|  |                +--rw direction?      identityref
|  |                +--rw policing?       identityref
|  |                +--rw byte-offset?    uint16
|  |                +--rw frame-delay
|  |                |  +--rw (flavor)?
|  |                |     +--:(lowest)
|  |                |     |  +--rw use-lowest-latency? empty
|  |                |     +--:(boundary)
|  |                |        +--rw delay-bound?     uint16
|  |                +--rw frame-jitter
|  |                |  +--rw (flavor)?
|  |                |     +--:(lowest)
|  |                |     |  +--rw use-lowest-jitter? empty
|  |                |     +--:(boundary)
|  |                |        +--rw delay-bound?     uint32
|  |                +--rw frame-loss
|  |                |  +--rw rate?   decimal64
|  |                +--rw bandwidth
|  |                   +--rw guaranteed-bw-percent decimal64
|  |                   +--rw end-to-end?           empty
|  +--rw carrierscarrier {carrierscarrier}?
|     +--rw signaling-type?   identityref
+--rw broadcast-unknown-unicast-multicast {bum}?
|  +--rw multicast-site-type?            enumeration
|  +--rw multicast-gp-address-mapping* [id]
|  |  +--rw id                 uint16
|  |  +--rw vlan-id            uint16
|  |  +--rw mac-gp-address     yang:mac-address
|  |  +--rw port-lag-number?   uint32
|  +--rw bum-overall-rate?     uint32
|  +--rw bum-rate-per-type* [type]
|     +--rw type    identityref
|     +--rw rate?   uint32
+--rw mac-loop-prevention {mac-loop-prevention}?
|  +--rw protection-type?   identityref
|  +--rw frequency?         uint32
|  +--rw retry-timer?       uint32
+--rw access-control-list
|  +--rw mac* [mac-address]
|     +--rw mac-address    yang:mac-address
+--ro actual-site-start?   yang:date-and-time
+--ro actual-site-stop?    yang:date-and-time
+--rw bundling-type?       identityref
+--rw default-ce-vlan-id   uint32
+--rw site-network-accesses
   +--rw site-network-access* [network-access-id]
   +--rw network-access-id                string
```

```
          +--rw remote-carrier-name?             string
          +--rw type?                            identityref
          +--rw (location-flavor)
          |  +--:(location)
          |  |  +--rw location-reference?
          |  |       -> ../../../locations/location/
          |  |               location-id
          |  +--:(device)
          |     +--rw device-reference?
          |          -> ../../../devices/device/device-id
          +--rw access-diversity {site-diversity}?
          |  +--rw groups
          |  |  +--rw group* [group-id]
          |  |     +--rw group-id    string
          |  +--rw constraints
          |     +--rw constraint* [constraint-type]
          |        +--rw constraint-type    identityref
          |        +--rw target
          |           +--rw (target-flavor)?
          |              +--:(id)
          |              |  +--rw group* [group-id]
          |              |     +--rw group-id    string
          |              +--:(all-accesses)
          |              |  +--rw all-other-accesses?   empty
          |              +--:(all-groups)
          |                 +--rw all-other-groups?     empty
          +--rw bearer
          |  +--rw requested-type {requested-type}?
          |  |  +--rw type?     string
          |  |  +--rw strict?   boolean
          |  +--rw always-on?         boolean {always-on}?
          |  +--rw bearer-reference?  string {bearer-reference}?
          +--rw connection
          |  +--rw encapsulation-type?    identityref
          |  +--rw eth-inf-type?          identityref
          |  +--rw tagged-interface
          |  |  +--rw type?               identityref
          |  |  +--rw dot1q-vlan-tagged {dot1q}?
          |  |  |  +--rw tg-type?    identityref
          |  |  |  +--rw cvlan-id    uint16
          |  |  +--rw priority-tagged
          |  |  |  +--rw tag-type?   identityref
          |  |  +--rw qinq {qinq}?
          |  |  |  +--rw tag-type?   identityref
          |  |  |  +--rw svlan-id    uint16
          |  |  |  +--rw cvlan-id    uint16
          |  |  +--rw qinany {qinany}?
          |  |  |  +--rw tag-type?   identityref
```

```
|  |  |  +--rw svlan-id    uint16
|  |  +--rw vxlan {vxlan}?
|  |     +--rw vni-id       uint32
|  |     +--rw peer-mode?   identityref
|  |     +--rw peer-list* [peer-ip]
|  |        +--rw peer-ip    inet:ip-address
|  +--rw untagged-interface
|  |  +--rw speed?                 uint32
|  |  +--rw mode?                  neg-mode
|  |  +--rw phy-mtu?               uint32
|  |  +--rw lldp?                  boolean
|  |  +--rw oam-802.3ah-link {oam-3ah}?
|  |  |  +--rw enable?   boolean
|  |  +--rw uni-loop-prevention?   boolean
|  +--rw lag-interfaces {lag-interface}?
|  |  +--rw lag-interface* [index]
|  |     +--rw index    string
|  |     +--rw lacp {lacp}?
|  |        +--rw enable?            boolean
|  |        +--rw mode?              neg-mode
|  |        +--rw speed?             uint32
|  |        +--rw mini-link-num?     uint32
|  |        +--rw system-priority?   uint16
|  |        +--rw micro-bfd {micro-bfd}?
|  |        |  +--rw enable?       enumeration
|  |        |  +--rw interval?     uint32
|  |        |  +--rw hold-timer?   uint32
|  |        +--rw bfd {bfd}?
|  |        |  +--rw enabled?        boolean
|  |        |  +--rw (holdtime)?
|  |        |     +--:(profile)
|  |        |     |  +--rw profile-name?
|  |        |     |    -> /l2vpn-svc/
|  |        |     |         vpn-profiles/
|  |        |     |        valid-provider-identifiers/
|  |        |     |       bfd-profile-identifier
|  |        |     +--:(fixed)
|  |        |        +--rw fixed-value?    uint32
|  |        +--rw member-links
|  |        |  +--rw member-link* [name]
|  |        |     +--rw name                string
|  |        |     +--rw speed?              uint32
|  |        |     +--rw mode?               neg-mode
|  |        |     +--rw link-mtu?           uint32
|  |        |     +--rw oam-802.3ah-link {oam-3ah}?
|  |        |        +--rw enable?   boolean
|  |        +--rw flow-control?      boolean
|  |        +--rw lldp?              boolean
```

```
|  +--rw cvlan-id-to-svc-map* [svc-id]
|  |  +--rw svc-id
|  |  |    -> /l2vpn-svc/vpn-services/vpn-service/
|  |  |          vpn-id
|  |  +--rw cvlan-id* [vid]
|  |     +--rw vid    uint16
|  +--rw l2cp-control {l2cp-control}?
|  |  +--rw stp-rstp-mstp?    control-mode
|  |  +--rw pause?            control-mode
|  |  +--rw lacp-lamp?        control-mode
|  |  +--rw link-oam?         control-mode
|  |  +--rw esmc?             control-mode
|  |  +--rw l2cp-802.1x?      control-mode
|  |  +--rw e-lmi?            control-mode
|  |  +--rw lldp?             boolean
|  |  +--rw ptp-peer-delay?   control-mode
|  |  +--rw garp-mrp?         control-mode
|  +--rw oam {oam}
|     +--rw md-name         string
|     +--rw md-level        uint16
|     +--rw cfm-802.1-ag* [maid]
|     |  +--rw maid                     string
|     |  +--rw mep-id?                  uint32
|     |  +--rw mep-level?               uint32
|     |  +--rw mep-up-down?             enumeration
|     |  +--rw remote-mep-id?           uint32
|     |  +--rw cos-for-cfm-pdus?        uint32
|     |  +--rw ccm-interval?            uint32
|     |  +--rw ccm-holdtime?            uint32
|     |  +--rw alarm-priority-defect?   identityref
|     |  +--rw ccm-p-bits-pri?       ccm-priority-type
|     +--rw y-1731* [maid]
|        +--rw maid                            string
|        +--rw mep-id?                         uint32
|        +--rw type?                        identityref
|        +--rw remote-mep-id?                  uint32
|        +--rw message-period?                 uint32
|        +--rw measurement-interval?           uint32
|        +--rw cos?                            uint32
|        +--rw loss-measurement?               boolean
|        +--rw synthethic-loss-measurement?    boolean
|        +--rw delay-measurement
|        |  +--rw enable-dm?   boolean
|        |  +--rw two-way?     boolean
|        +--rw frame-size?                     uint32
|        +--rw session-type?                enumeration
+--rw availability
|  +--rw access-priority?   uint32
```

```
|  +--rw (redundancy-mode)?
|     +--:(single-active)
|     |  +--rw single-active?     empty
|     +--:(all-active)
|        +--rw all-active?        empty
+--rw vpn-attachment
|  +--rw (attachment-flavor)
|     +--:(vpn-id)
|     |  +--rw vpn-id?
|     |  |    -> /l2vpn-svc/vpn-services/
|     |  |            vpn-service/vpn-id
|     |  +--rw site-role?              identityref
|     +--:(vpn-policy-id)
|        +--rw vpn-policy-id?
|             -> ../../../../vpn-policies/
|                     vpn-policy/vpn-policy-id
+--rw service
|  +--rw svc-bandwidth {input-bw}?
|  |  +--rw bandwidth* [direction type]
|  |     +--rw direction    identityref
|  |     +--rw type         identityref
|  |     +--rw cos-id?      uint8
|  |     +--rw vpn-id?      svc-id
|  |     +--rw cir          uint64
|  |     +--rw cbs          uint64
|  |     +--rw eir?         uint64
|  |     +--rw ebs?         uint64
|  |     +--rw pir?         uint64
|  |     +--rw pbs?         uint64
|  +--rw svc-mtu            uint16
|  +--rw qos {qos}?
|  |  +--rw classification-policy
|  |  |  +--rw rule* [id]
|  |  |     +--rw id                    string
|  |  |     +--rw (match-type)?
|  |  |     |  +--:(match-flow)
|  |  |     |  |  +--rw match-flow
|  |  |     |  |     +--rw dscp?           inet:dscp
|  |  |     |  |     +--rw dot1q?          uint16
|  |  |     |  |     +--rw pcp?            uint8
|  |  |     |  |     +--rw src-mac?  yang:mac-address
|  |  |     |  |     +--rw dst-mac?  yang:mac-address
|  |  |     |  |     +--rw color-type?     identityref
|  |  |     |  |     +--rw target-sites*
|  |  |     |  |     |          svc-id {target-sites}?
|  |  |     |  |     +--rw any?            empty
|  |  |     |  |     +--rw vpn-id?         svc-id
|  |  |     |  +--:(match-application)
```

```
|  |  |     |     +--rw match-application? identityref
|  |  |     +--rw target-class-id?     string
|  |  +--rw qos-profile
|  |     +--rw (qos-profile)?
|  |        +--:(standard)
|  |        |  +--rw profile?
|  |        |       -> /l2vpn-svc/vpn-profiles/
|  |        |             valid-provider-identifiers/
|  |        |             qos-profile-identifier
|  |        +--:(custom)
|  |           +--rw classes {qos-custom}?
|  |              +--rw class* [class-id]
|  |                 +--rw class-id        string
|  |                 +--rw direction?      identityref
|  |                 +--rw policing?       identityref
|  |                 +--rw byte-offset?    uint16
|  |                 +--rw frame-delay
|  |                 |  +--rw (flavor)?
|  |                 |     +--:(lowest)
|  |                 |     |  +--rw use-lowest-latency?
|  |                 |     |                     empty
|  |                 |     +--:(boundary)
|  |                 |        +--rw delay-bound? uint16
|  |                 +--rw frame-jitter
|  |                 |  +--rw (flavor)?
|  |                 |     +--:(lowest)
|  |                 |     |  +--rw use-lowest-jitter?
|  |                 |     |                     empty
|  |                 |     +--:(boundary)
|  |                 |        +--rw delay-bound? uint32
|  |                 +--rw frame-loss
|  |                 |  +--rw rate?   decimal64
|  |                 +--rw bandwidth
|  |                    +--rw guaranteed-bw-percent
|  |                    |                    decimal64
|  |                    +--rw end-to-end?       empty
|  +--rw carrierscarrier {carrierscarrier}?
|     +--rw signaling-type?   identityref
+--rw broadcast-unknown-unicast-multicast {bum}?
|  +--rw multicast-site-type?           enumeration
|  +--rw multicast-gp-address-mapping* [id]
|  |  +--rw id                 uint16
|  |  +--rw vlan-id            uint16
|  |  +--rw mac-gp-address     yang:mac-address
|  |  +--rw port-lag-number?   uint32
|  +--rw bum-overall-rate?              uint32
|  +--rw bum-rate-per-type* [type]
|     +--rw type    identityref
```

```
|     +--rw rate?   uint32
+--rw mac-loop-prevention {mac-loop-prevention}?
|  +--rw protection-type?   identityref
|  +--rw frequency?         uint32
|  +--rw retry-timer?       uint32
+--rw access-control-list
|  +--rw mac* [mac-address]
|     +--rw mac-address    yang:mac-address
+--rw mac-addr-limit
+--rw limit-number?    uint16
+--rw time-interval?   uint32
+--rw action?          identityref
```

Figure 4

5.1. Features and Augmentation

The model defined in this document implements many features that allow implementations to be modular. As an example, the layer 2 protocols parameters (Section 5.3.2.2) proposed to the customer may also be enabled through features. This model also defines some features for options that are more advanced, such as support for extranet VPNs (Section 5.2.4), site diversity (Section 5.3), and QoS (Section 5.10.2).

In addition, as for any YANG model, this service model can be augmented to implement new behaviors or specific features. For example, this model defines VXLAN [RFC7348] for Ethernet packet Encapsulation; if VXLAN Encapsulation does not fulfill all requirements for describing the service, new options can be added through augmentation.

5.2. VPN Service Overview

A vpn-service list item contains generic information about the VPN service. The vpn-id of the vpn-service refers to an internal reference for this VPN service. This identifier is purely internal to the organization responsible for the VPN service.

A vpn-service is composed of some characteristics:

Customer information: Used to identify the customer.

VPN Service Type (svc-type): Used to indicate the VPN service Type. The identifier is an identity allowing any encoding for the local administration of the VPN service. Note that other identity can be an extension of the base identity.

Cloud Access (cloud-access): All sites in the L2VPN SHOULD be permitted to access to the cloud by default. The cloud-access container provides parameters for authorization rules. A cloud identifier is used to reference the target service. This identifier is local to each administration.

Service Topology (svc-topo): Used to identify the type of VPN service topology that is required.

Frame Delivery Service (frame-delivery): Defines the frame delivery support required for the L2VPN, e.g., multicast delivery, unicast delivery, or broadcast delivery.

Extranet VPN (extranet-vpns): Indicates that a particular VPN needs access to resources located in another VPN.

5.2.1. VPN Service Type

The "vpn-svc-type" defines the service type for provider provisioned L2VPNs. The current version of the model supports six flavors:

- Point-to-point Virtual Private Wire Services (VPWS) connecting two customer Sites;
- Point-to-point or point-to-multipoint Virtual Private Wire Services (VPWS) connecting a set of customer sites [RFC8214];
- Multipoint Virtual Private LAN services (VPLS) connecting a set of customer sites;
- Multipoint Virtual Private LAN services (VPLS) connecting one or more root sites and a set of leaf sites, but preventing inter-leaf sites communication.
- EVPN Service connecting a set of customer sites.
- Ethernet VPN VPWS between two customer sites or a set of customer sites specified in [RFC8214] and [RFC7432];

Other L2VPN Service Types could be included by augmentation. Note that an Ethernet Private Line (EPL) service or an Ethernet Virtual Private Line (EVPL) service is an E-Line service [MEF-6]or a point-to-point Ethernet Virtual Circuit (EVC) service, while an Ethernet Private LAN (EP-LAN) service or an Ethernet Virtual Private LAN (EVP-LAN) service is an E-LAN service [MEF-6] or a multipoint-to-multipoint EVC service.

5.2.2. VPN Service Topology

The type of VPN service topology can be used for configuration if needed. The module currently supports: any-to-any; hub-and-spoke (where hubs can exchange traffic); and hub-and-spoke-disjoint (where hubs cannot exchange traffic). New topologies could be added by augmentation. By default, the any-to-any VPN service topology is used.

5.2.2.1. Route Target Allocation

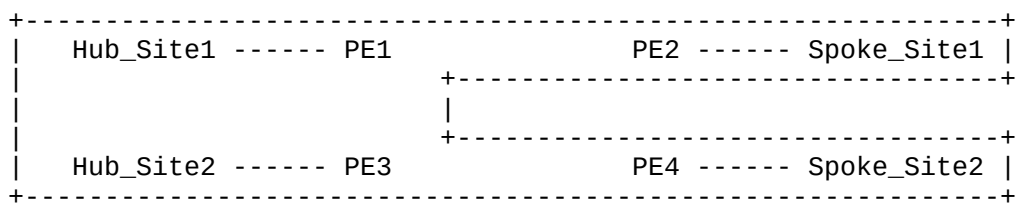
A Layer 2 PE-based VPN (such as a VPLS-based VPN, or an EVPN that uses BGP as its signaling protocol) can be built using route targets (RTs) as described in [RFC4364] an [RFC7432]. The management system is expected to automatically allocate a set of RTs upon receiving a VPN service creation request. How the management system allocates RTs is out of scope for this document, but multiple ways could be envisaged, as described in the Section 6.2.1.1 of [RFC8299].

5.2.2.2. Any-to-Any

Any-to-Any VPN Service Topology

In the any-to-any VPN service topology, all VPN sites can communicate with each other without any restrictions. The management system that receives an any-to-any L2VPN service request through this model is expected to assign and then configure the MAC-VRF and RTs on the appropriate PEs. In the any-to-any case, a single RT is generally required, and every MAC-VRF imports and exports this RT.

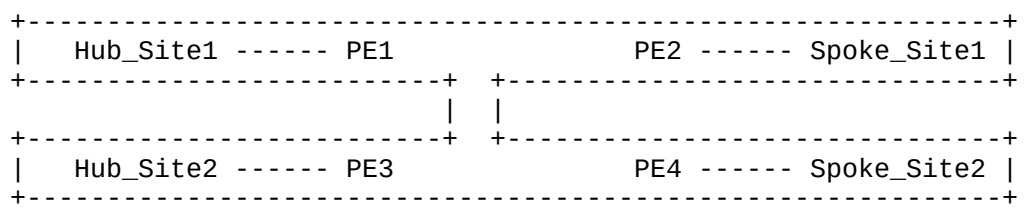
5.2.2.3. Hub-and-Spoke



Hub-and-Spoke VPN Service Topology

In the Hub-and-Spoke VPN service topology, all Spoke sites can communicate only with Hub sites, but not with each other. And Hubs can also communicate with each other. The management system that receives a Hub-and-Spoke L2VPN service request through this model is expected to assign and then configure the MAC-VRF and RTs on the appropriate PEs. In the Hub-and-Spoke case, two RTs are generally required (one RT for Hub routes, and one RT for Spoke routes). A Hub MAC-VRF that connects Hub sites will export Hub routes with the Hub RT and will import Spoke routes through the Spoke RT. It will also import the Hub RT to allow Hub-to-Hub communication. A Spoke MAC-VRF that connects Spoke sites will export Spoke routes with the Spoke RT and will import Hub routes through the Hub RT.

5.2.2.4. Hub-and-Spoke-Disjoint



Hub-and-Spoke-Disjoint VPN Service Topology

In the Hub-and-Spoke-Disjoint VPN service topology, all Spoke sites can communicate only with Hub sites, but not with each other. And Hubs cannot communicate with each other. The management system that receives a Hub-and-Spoke-Disjoint L2VPN service request through this model is expected to assign and then configure the VRF and RTs on the appropriate PEs. In the Hub-and-Spoke-Disjoint case, at least two RTs are required for Hub and Spoke respectively(one RT for Hub routes and one RT for Spoke routes). A Hub VRF that connects Hub sites will export Hub routes with the Hub RT and will import Spoke routes through the Spoke RT. A Spoke VRF that connects Spoke sites will export Spoke routes with the Spoke RT and will import Hub routes through the Hub RT.

The management system MUST take into account constraints on Hub-and-Spoke connections, as in the previous case.

Hub-and-Spoke-Disjoint can also be seen as multiple Hub-and-Spoke VPNs (one per Hub) that share a common set of Spoke sites.

5.2.3. Cloud Access

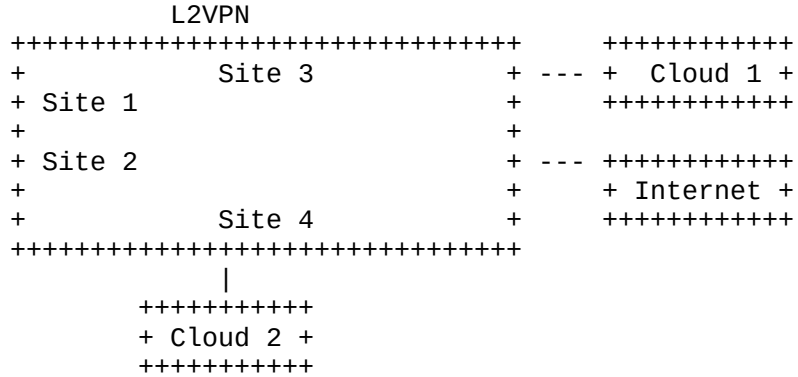
This model provides cloud access configuration through the cloud-access container. The usage of cloud-access is targeted for public cloud and for Internet access. The cloud-access container provides parameters for authorization rules. Note that this model considers that public cloud and public Internet access share some commonality, therefore it does not distinguish Internet access from cloud access. Anyway, a different label for Internet access could be added by augmentation.

Private cloud access may be addressed through the site container as described in Section 5.3 with use consistent with sites of type NNI (Network to Network Interface).

A cloud identifier is used to reference the target service. This identifier is local to each administration.

By default, all sites in the L2VPN SHOULD be permitted to access the cloud or internet. If restrictions are required, a user MAY configure some limitations for some sites or nodes by using policies, i.e. the "permit-site" or "deny-site" leaf-list. The permit-site leaf-list defines the list of sites authorized for cloud access. The deny-site leaf-list defines the list of sites denied for cloud access. The model supports both "deny-any-except" and "permit-any-except" authorization.

How the restrictions will be configured on network elements is out of scope for this document.



In the example above, we configure the global VPN to access the Internet by creating a cloud-access container pointing to the cloud identifier for the Internet service. No authorized sites will be

configured, as all sites are required to be able to access the Internet.

```
 <?xml version="1.0"?>
    <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
    <vpn-services>
    <vpn-service>
    <vpn-id>123456487</vpn-id>
   <cloud-accesses>
    <cloud-access>
       <cloud-identifier>INTERNET</cloud-identifier>
    </cloud-access>
   </cloud-accesses>
   <ce-vlan-preservation>true</ce-vlan-preservation>
   <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
   </vpn-service>
   </vpn-services>
 </l2vpn-svc>
```

If Site 1 and Site 2 require access to Cloud 1, a new cloud-access container pointing to the cloud identifier of Cloud 1 will be created. The permit-site leaf-list will be filled with a reference to Site 1 and Site 2.

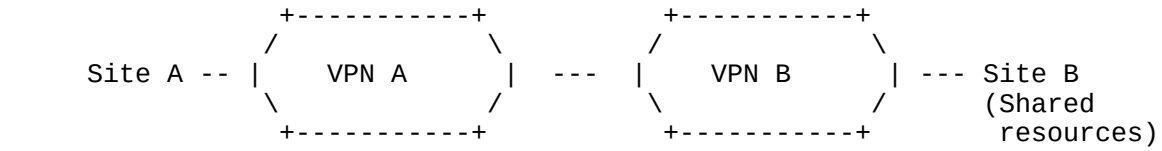
```
 <?xml version="1.0"?>
    <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
     <vpn-services>
      <vpn-service>
      <vpn-id>123456487</vpn-id>
      <cloud-accesses>
       <cloud-access>
         <cloud-identifier>Cloud1</cloud-identifier>
         <permit-site>site1</permit-site>
         <permit-site>site2</permit-site>
       </cloud-access>
      </cloud-accesses>
     <ce-vlan-preservation>true</ce-vlan-preservation>
     <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
    </vpn-service>
    </vpn-services>
 </l2vpn-svc>
```

If all sites except Site 1 require access to Cloud 2, a new cloud-access container pointing to the cloud identifier of Cloud 2 will be created. The deny-site leaf-list will be filled with a reference to Site 1.

```
<?xml version="1.0"?>
  <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
    <vpn-services>
     <vpn-service>
       <vpn-id>123456487</vpn-id>
        <cloud-accesses>
         <cloud-access>
          <cloud-identifier>Cloud2</cloud-identifier>
          <deny-site>site1</deny-site>
        </cloud-access>
       </cloud-accesses>
      <ce-vlan-preservation>true</ce-vlan-preservation>
      <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
    </vpn-service>
  </vpn-services>
</l2vpn-svc>
```

5.2.4. Extranet VPNs

There are some cases where a particular VPN needs access to resources (servers, hosts, etc.) that are external. Those resources may be located in another VPN.



In the figure above, VPN B has some resources on Site B that need to be made available to some customers/partners. Specifically, VPN A must be able to access those VPN B resources.

Such a VPN connection scenario can be achieved via a VPN policy as defined in Section 5.5.2.2. But there are some simple cases where a particular VPN (VPN A) needs access to all resources in another VPN (VPN B). The model provides an easy way to set up this connection using the "extranet-vpns" container.

The extranet-vpns container defines a list of VPNs a particular VPN wants to access. The extranet-vpns container is used on customer VPNs accessing extranet resources in another VPN. In the figure above, in order to provide VPN A with access to VPN B, the extranet-vpns container needs to be configured under VPN A with an entry corresponding to VPN B. There is no service configuration requirement on VPN B.

Readers should note that even if there is no configuration requirement on VPN B, if VPN A lists VPN B as an extranet, all sites in VPN B will gain access to all sites in VPN A.

The "site-role" leaf defines the role of the local VPN sites in the target extranet VPN service topology. Site roles are defined in Section 5.4.

In the example below, VPN A accesses VPN B resources through an extranet connection. A Spoke role is required for VPN A sites, as sites from VPN A must not be able to communicate with each other through the extranet VPN connection.

```
<?xml version="1.0"?>
  <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
   <vpn-services>
     <vpn-service>
       <vpn-id>VPNB</vpn-id>
         <svc-topo>hub-spoke</svc-topo>
        <ce-vlan-preservation>true</ce-vlan-preservation>
        <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
     </vpn-service>
     <vpn-service>
        <vpn-id>VPNA</vpn-id>
          <svc-topo>any-to-any</svc-topo>
             <extranet-vpns>
               <extranet-vpn>
                <vpn-id>VPNB</vpn-id>
                <local-sites-role>spoke-role</local-sites-role>
              </extranet-vpn>
            </extranet-vpns>
        <ce-vlan-preservation>true</ce-vlan-preservation>
        <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
    </vpn-service>
   </vpn-services>
  </l2vpn-svc>
```

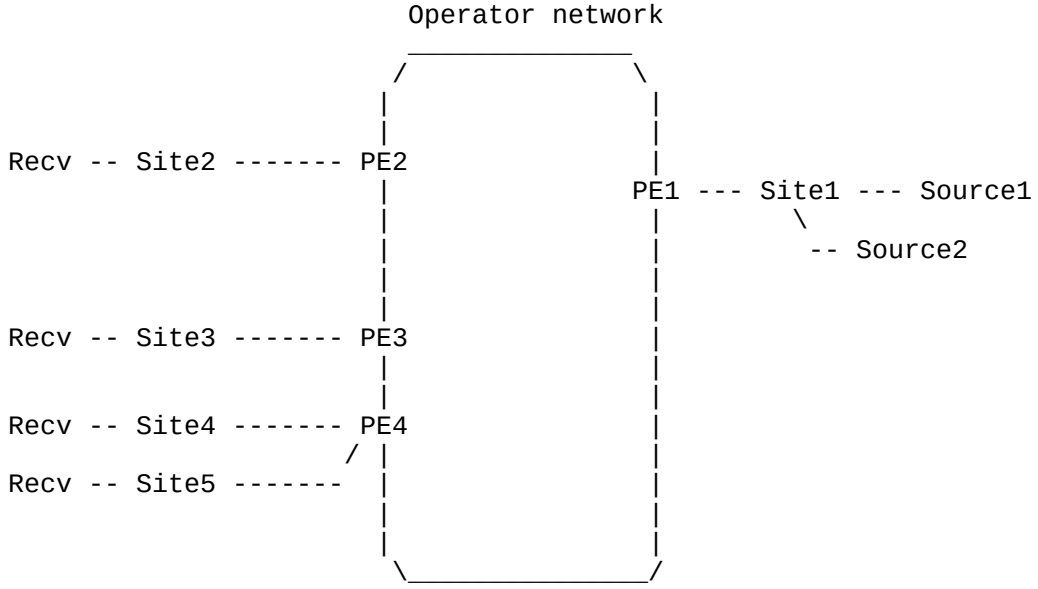
This model does not define how the extranet configuration will be achieved within the network.

Any VPN interconnection scenario that is more complex (e.g., only certain parts of sites on VPN A accessing only certain parts of sites on VPN B) needs to be achieved using a VPN attachment as defined in Section 5.5.2, and especially a VPN policy as defined in Section 5.5.2.2.

5.2.5. Frame Delivery Service

If BUM (Broadcast/Unknown/Multicast) Frame Delivery Service is supported for an L2VPN, some global frame delivery parameters are required as input for the service request. When a CE sends packets that are Broadcast, Multicast, or Unknown-destination-unicast, replication occurs at the ingress PE, three frame types need to be supported.

Users of this model will need to provide the flavors of trees that will be used by customers within the L2VPN (customer-tree-flavors). The model defined in this document supports bidirectional, shared, and source-based trees (and can be augmented to contain other tree types). Multiple flavors of trees can be supported simultaneously.

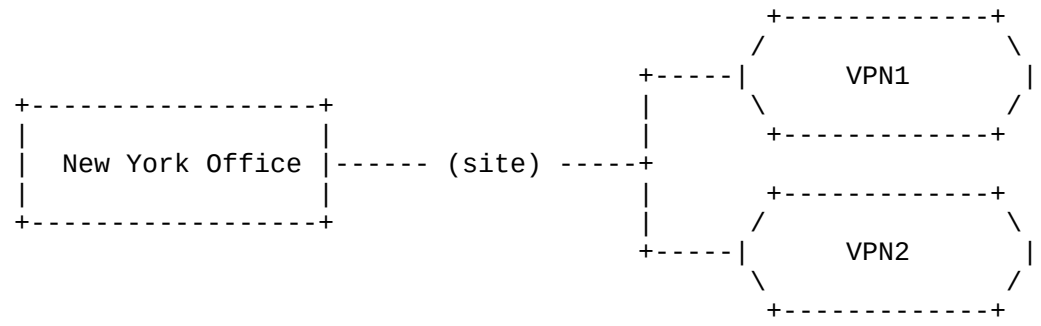


Multicast Group to port mappings can be created using the "rp-group-mappings" leaf. Two group to port mapping methods are supported:

- Static configuration of multicast Ethernet addresses and ports/ interfaces.
- Multicast control protocol based on Layer-2 technology that signals mappings of multicast addresses to ports/interfaces, such as Generic Attribute Registration Protocol / GARP Multicast Registration Protocol (GARP/GMRP) [IEEE-802-1D].

5.3. Site Overview

A site represents a connection of a customer office to one or more VPN services. Each site is associated with one or more location.



The "site" container is used for the provider to store information of detailed implementation arrangements made with either the customer or with peer operators at each inter-connect location.

We restrict the L2SM to exterior interfaces (i.e.,UNI and NNI) only, so all internal interfaces and the underlying topology are outside the scope of L2SM.

Typically, the following characteristics of a site interface handoff need to be documented as part of the service design:

Unique identifier (site-id): An arbitrary string to uniquely identify the site within the overall network infrastructure. The format of site-id is determined by the local administration of the VPN service.

Device (device): The customer can request one or more customer premise equipments from the service provider for a particular site.

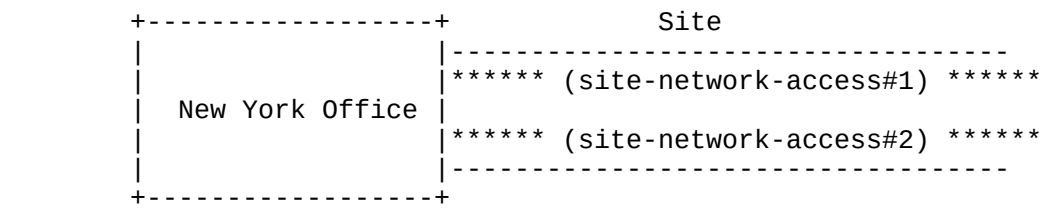
Management (management): Defines the model of management for the site, for example: type, management-transport, address. This decides the boundary between service provider and customer: who has ownership of CE device.

Location (location): The site location information to allow easy retrieval of data about which are the nearest available resources.

Site diversity (site-diversity): Presents some parameters to support site diversity.

Site Network Accesses (site-network-accesses): Defines the list of ports to the site and their properties: especially bearer, connection, and service parameters.

A site-network-access represents an Ethernet logical connection to a site. A site may have multiple site-network-accesses.



Multiple site-network-accesses are used, for instance, in the case of multihoming. Some other meshing cases may also include multiple site-network-accesses.

The site configuration is viewed as a global entity; we assume that it is mostly the management system's role to split the parameters between the different elements within the network. For example, in the case of the site-network-access configuration, the management system needs to split the parameters between the PE configuration and the CE configuration.

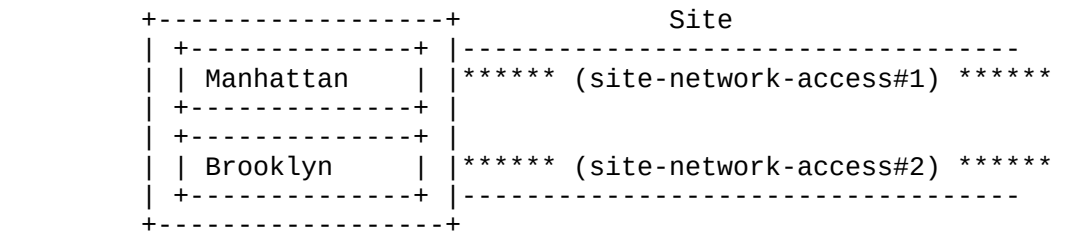
The site may support single-homed acess or multihoming. In the case of multihoming, the site can support multiple site-network-accesses, under each site-network-access, vpn-attachment is defined and it will describe which site-network-access associated with which site will connect to which VPN.

5.3.1. Devices and Locations

The information in the "location" sub-container under a "site" and in the "device" container allows easy retrieval of data about which are the nearest available facilities and can be used for access topology planning. It may also be used by other network orchestration components to choose the targeted upstream PE and downstream CE. Location is expressed in terms of postal information. More detailed or other location information can be added by augmentation.

A site may be composed of multiple locations. All the locations will need to be configured as part of the "locations" container and list. A typical example of a multi-location site is a headquarters office in a city composed of multiple buildings. Those buildings may be located in different parts of the city and may be linked by intra-city fibers (a customer metropolitan area network). This model does

not represent the connectivity between the multiple locations of a site, because that connectivity is controlled by the customer. In such a case, when connecting to a VPN service, the customer may ask for multihoming based on its distributed locations.



A customer may also request the use of some premises equipment entities (CEs) from the SP via the "devices" container. Requesting a CE implies a provider-managed or co-managed model. A particular device must be ordered to a particular already-configured location. This would help the SP send the device to the appropriate postal address. In a multi-location site, a customer may, for example, request a CE for each location on the site where multihoming must be implemented. In the figure above, one device may be requested for the Manhattan location and one other for the Brooklyn location.

By using devices and locations, the user can influence the multihoming scenario he wants to implement: single CE, dual CE, etc.

5.3.2. Site Network Accesses

The L2SM includes a set of essential physical interface properties and Ethernet layer characteristics in the "site-network-accesses" container. Some of these are critical implementation arrangements that require consent from both customer and provider.

As mentioned earlier, a site may be multihomed. Each logical network access for a site is defined in the "site-network-accesses" container. The site-network-access parameter defines how the site is connected on the network and is split into three main classes of parameters:

- bearer: defines requirements of the attachment (below Layer 2).
- connection: defines Layer 2 protocol parameters of the attachment.
- availability: defines the site’s availability policy. The availability parameters are defined in Section 5.8.

The site-network-access has a specific type (site-network-access-type). This document defines two types:

- point-to-point: describes a point-to-point connection between the SP and the customer.
- multipoint: describes a multipoint connection between the SP and the customer.

This site-network-access type may have an impact on the parameters offered to the customer, e.g., an SP might not offer MAC Loop Protection for multipoint accesses. It is up to the provider to decide what parameters are supported for point-to-point and/or multipoint accesses. Multipoint accesses are out of scope for this document and some containers defined in the model may require extensions in order to work properly for multipoint accesses.

5.3.2.1. Bearer

The "bearer" container defines the requirements for the site attachment to the provider network that are below Layer 2.

The bearer parameters will help to determine the access media to be used.

5.3.2.2. Connection

The "connection" container defines the layer 2 protocol parameters of the attachment (e.g.,vlan-id or circuit-id) and provides connectivity between customer Ethernet switches. Depending on the management mode, it refers to PE-CE-LAN segment addressing or to CE-to-customer-LAN segment addressing. In any case, it describes the responsibility boundary between the provider and the customer. For a customer-managed site, it refers to the PE-CE-LAN Segment connection. For a provider-managed site, it refers to the CE-to-LAN Segment connection.

"encapsulation-type" allows the user to select between Ethernet encapsulation (port-based) or Ethernet VLAN encapsulation (VLAN-based). All of the allowed Ethernet interface types of service frame can be listed under "ether-inf-type", e.g., untagged interface, tagged interface, LAG interface.

Corresponding to "ether-inf-type", the connection container also presents three sets of link attributes: untagged interface, tagged interface, or optional LAG interface attributes. These parameters are essential for the connection to establish properly between the customer and provider edge devices. The connection container also

defines an L2CP attribute to allow control plane protocol interaction between the CE devices and PE device.

5.3.2.2.1. Untagged Interface

For each untagged interface (untagged-interface), there are basic configuration parameters like interface index and speed, interface MTU, auto-negotiation and flow-control settings, etc. In addition and based on mutual agreement, the customer and provider may decide to enable advanced features, such as LLDP, IEEE 802.3ah, MAC loop detection/prevention at a UNI. If Loop avoidance is required, the attribute "uni-loop-prevention" must be set to TRUE.

5.3.2.2.2. Tagged Interface

If the tagged service is enabled on a logical unit on the connection at the interface, "encapsulation-type" should be specified as the Ethernet VLAN ecapsulation (if VLAN-based) or VXLAN encapsulation, and "eth-inf-type" should be set to indicate a tagged interface.

In addition, "tagged-interface-type" should be specified under "tagged-interface" container to determines how tagging needs to be done. The current model defines five ways to perform VLAN tagging:

- priority-tagged: Service providers encapsulate and tag packets between CE and PE with the frame priority level.

- dot1q-vlan-tagged: Service providers encapsulate packets between CE and PE with one or a set of customer VLAN IDs (C-VLANs)

- qinq: service providers encapsulate packets that enter the service-provider network with multiple customer VLAN IDs (C-VLANs) and a single VLAN tag with a single service provider VLAN (S-VLAN).

- qinany: service providers encapsulate packets that enter the service-provider network with unknown C-VLAN and a single VLAN tag with a single service provider VLAN (S-VLAN).

- vxlan: service providers encapsulate packets that enter the service-provider network with VNI and peer list.

The overall S-tag for the Ethernet circuit and C-tag-to-SVC mapping, if applicable, has been placed in the service container. For the qinq an qinany options, the S-tag under "qinq" and "qinany" should match the S-tag in the service container in most cases, however, vlan translation is required for the S-tag in certain deployment at the external facing interface or upstream PEs to "normalize" the outer

VLAN tag to the service S-tag into the network and translate back to the site's S-tag in the opposite direction. One example of this is with a Layer 2 aggregation switch along the path: the S-tag for the SVC has been previously assigned to another service thus can not be used by this attachment circuit.

5.3.2.2.3. LAG Interface

Sometimes, the customer may require multiple physical links bundled together to form a single, logical, point-to-point LAG connection to the service provider. Typically, LACP (Link Aggregation Control Protocol) is used to dynamically manage adding or deleting member links of the aggregate group. In general, a LAG allows for increased service bandwidth beyond the speed of a single physical link while providing graceful degradation as failure occurs, thus increased availability.

In the L2SM, there is a set of attributes under "LAG-interface" related to link aggregation functionality. The customer and provider first need to decide on whether LACP PDUs will be exchanged between the edge devices by specifying the "LACP-state" to "On" or "Off". If LACP is to be enabled, then both parties need to further specify whether it will be running in active or passive mode, plus the time interval and priority level of the LACP PDU. The customer and provider can also determine the minimum aggregate bandwidth for a LAG to be considered as a valid path by specifying the optional "mini-link" attribute. To enable fast detection of faulty links, micro-BFD [RFC7130]runs independent UDP sessions to monitor the status of each member link. Customer and provider should agree the BFD hello interval and hold time.

Each member link will be listed under the LAG interface with basic physical link properties. Certain attributes like flow-control, encapsulation type, allowed ingress Ethertype and LLDP settings are at the LAG level.

5.3.2.2.4. CVLAN ID To SVC MAP

When more than one service is multiplexed onto the same interface, ingress service frames are conditionally transmitted through one of the L2VPN services based upon pre-arranged customer VLAN-to-SVC mapping. Multiple customer VLANs can be bundled across the same SVC. The bundling type will determine how a group of CVLANs is bundled into one VPN service (i.e., VLAN-Bundling).

"cvlan-id-to-svc-map", when applicable, contains the list of customer VLANs that are mapped to the same service. In most cases, this will be the VLAN access-list for the inner 802.1q tag (the C-tag).

A VPN Service can be set to preserve the CE-VLAN ID and CE-VLAN CoS from source site to destination site. This is required when the customer wants to use the VLAN header information between its two sites. CE-VLAN ID Preservation and CE-VLAN CoS Preservation are applied on each site-network-access within sites. Preservation means that the value of CE-VLAN ID and/or CE-VLAN CoS at source site must be equal to the value at a destination site belonging to the same L2VPN Service.

If All-to-One bundling is Enabled (i.e., bundling type is set to all-to-one bundling), then preservation applies to all Ingress service frames. If All-to-One bundling is Disabled, then preservation applies to tagged Ingress service frames having the CE-VLAN ID.

5.3.2.2.5. L2CP Control Support

Customer and Service provider should pre-arrange whether to allow control plane protocol interaction between the CE devices and PE device. To provide seamless operation with multicast data transport, the transparent operation of Ethernet control protocols (e.g., Spanning Tree Protocol [IEEE-802-1D]) can be employed by customers.

To support efficient dynamic transport, Ethernet multicast control frames (e.g., GARP/GMRP [IEEE-802-1D]) can be used between CE and PE. However, solutions MUST NOT assume all CEs are always running such protocols (typically in the case where a CE is a router and is not aware of Layer-2 details).

The destination MAC addresses of these L2CP PDUs fall within two reserved blocks specified by the IEEE 802.1 Working Group. Packet with destination MAC in these multicast ranges have special forwarding rules.

- Bridge Block of Protocols: 01-80-C2-00-00-00 through 01-80-C2-00-00-0F

- MRP Block of Protocols: 01-80-C2-00-00-20 through 01-80-C2-00-00-2F

Layer 2 protocol tunneling allows service providers to pass subscriber Layer 2 control PDUs across the network without being interpreted and processed by intermediate network devices. These L2CP PDUs are transparently encapsulated across the MPLS-enabled core network in Q-in-Q fashion.

The "L2CP-control" container contains the list of commonly used L2CP protocols and parameters. The service provider can specify DISCARD, PEER, or TUNNEL mode actions for each individual protocol.

5.3.2.2.6. Ethernet Service OAM

The advent of Ethernet as a wide-area network technology brings additional requirements of end-to-end service monitoring and fault management in the SP network, particularly in the area of service availability and Mean Time To Repair (MTTR). Ethernet Service OAM in the L2SM model refers to the combined protocol suites of IEEE 802.1ag ([IEEE-802-1ag]) and ITU-T Y.1731 ([ITU-T-Y-1731]).

Generally speaking, Ethernet Service OAM enables service providers to perform service continuity check, fault-isolation, and packet delay/ jitter measurement at per-customer and per-site network access granularity. The information collected from Ethernet Service OAM data sets is complementary to other higher layer IP/MPLS OSS tools to ensure the required service level agreements (SLAs) can be meet.

The 802.1ag Connectivity Fault Management (CFM) functional model is structured with hierarchical maintenance domains (MDs), each assigned with a unique maintenance level. Higher level MDs can be nested over lower level MDs. However, the MDs cannot intersect. The scope of each MD can be solely within a customer network, solely within the SP network, interact between the customer-to-provider or provider-to-provider edge equipment, or tunnel over another SP network.

Depending on the use case scenario, one or more maintenance end points (MEPs) can be placed on the external facing interface, sending CFM PDUs towards the core network (Up MEP) or downstream link (Down MEP).

The "cfm-802.1-ag" sub-container under "site-network-access" presents the CFM maintenance association (MA): i.e., Down MEP for the UNI MA. For each MA, the user can define the maintenance domain ID (MAID), MEP level, MEP direction, remote MEP ID, CoS level of the CFM PDUs, Continuity Check Message (CCM) interval and hold time, alarm priority defect, CCM priority-type, etc.

ITU-T Y.1731 Performance Monitoring (PM) provides essential network telemetry information that includes the measurement of Ethernet service frame delay, frame delay variation, frame loss, and frame throughput. The delay/jitter measurement can be either one-way or two-way. Typically, a Y.1731 PM probe sends a small amount of synthetic frames along with service frames to measure the SLA parameters.

The "y-1731" sub-container under "site-network-access" contains a set of parameters to define the PM probe information, including MAID, local and remote MEP-ID, PM PDU type, message period and measurement interval, CoS level of the PM PDUs, loss measurement by synthetic or

service frame options, one-way or two-way delay measurement, PM frame size, and session type.

5.4. Site Role

A VPN has a particular service topology, as described in Section 5.2.2. As a consequence, each site belonging to a VPN is assigned a particular role in this topology. The site-role leaf defines the role of the site in a particular VPN topology.

In the any-to-any VPN service topology, all sites MUST have the same role, which will be "any-to-any-role".

In the Hub-and-Spoke VPN service topology or the Hub-and-Spoke-Disjoint VPN service topology, sites MUST have a Hub role or a Spoke role.

5.5. Site Belonging to Multiple VPNs

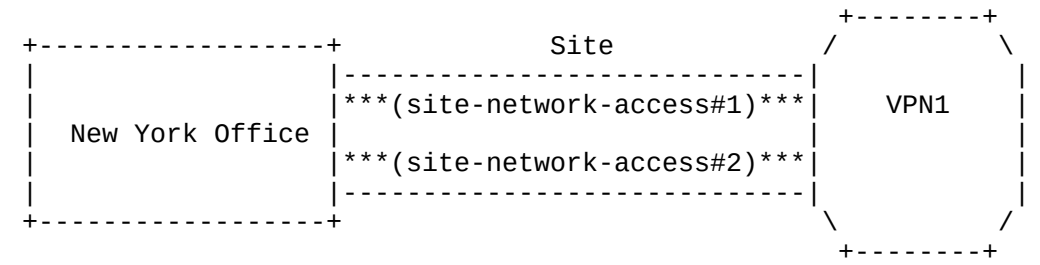
5.5.1. Site VPN Flavor

A site may be part of one or more VPNs. The "site-vpn-flavor" defines the way the VPN multiplexing is done. There are four possible types of external facing connections associated with an Ethernet VPN service and a site. Therefore the model supports four flavors:

- site-vpn-flavor-single: The site belongs to only one VPN.
- site-vpn-flavor-multi: The site belongs to multiple VPNs, and all the logical accesses of the sites belong to the same set of VPNs.
- site-vpn-flavor-nni: The site represents an NNI where two administrative domains belonging to the same or different providers inter-connect.
- site-vpn-flavor-e2e: The site represents an end-to-end multi-segment connection.

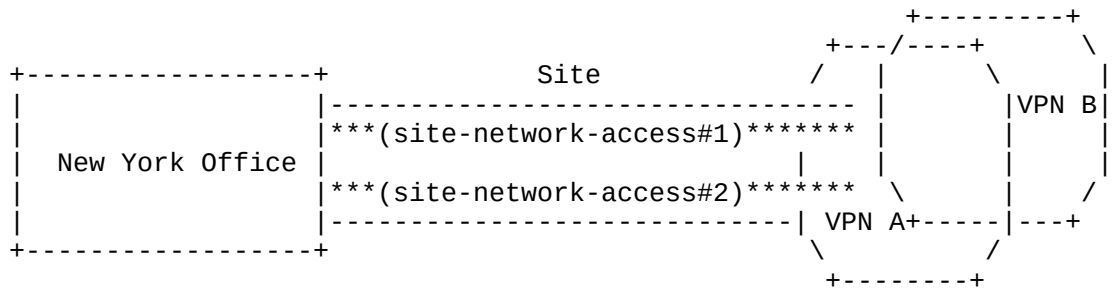
5.5.1.1. Single VPN Attachment: site-vpn-flavor-single

The figure below describes a single VPN attachment. The site connects to only one VPN.



5.5.1.2. MultiVPN Attachment: site-vpn-flavor-multi

The figure below describes a site connected to multiple VPNs.

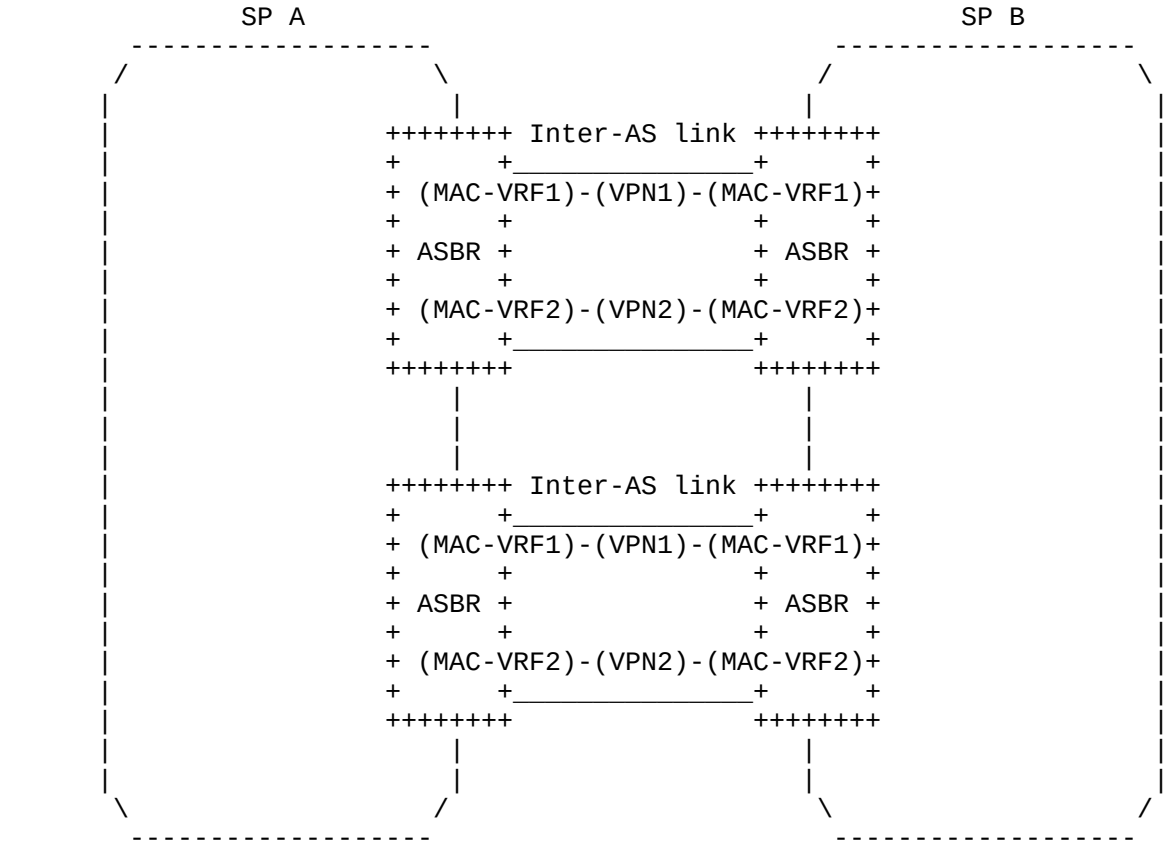


In the example above, the New York office is multihomed. Both logical accesses are using the same VPN attachment rules, and both are connected to VPN A and to VPN B.

Reaching VPN A or VPN B from the New York office will be done via MAC destination based forwarding. Having the same destination reachable from the two VPNs may cause routing problems. The customer administration's role in this case would be to ensure the appropriate mapping of its MAC addresses in each VPN. See Section 5.5.2 and Section 5.10.2 for more details. See also Section 5.10.3 for BUM support.

5.5.1.3. NNI: site-vpn-flavor-nni

A Network-to-Network Interface (NNI) scenario may be modeled using the sites container. It is helpful for the SP to indicate that the requested VPN connection is not a regular site but rather is an NNI, as specific default device configuration parameters may be applied in the case of NNIs (e.g., ACLs, routing policies).



The figure above describes an option A NNI scenario that can be modeled using the sites container. In order to connect its customer VPNs (VPN1 and VPN2) in SP B, SP A may request the creation of some site-network-accesses to SP B. The site-vpn-flavor-nni will be used to inform SP B that this is an NNI and not a regular customer site.

5.5.1.4. E2E: site-vpn-flavor-e2e

A end to end multi-segment VPN connection to be constructed out of several connectivity segments may be modeled. It is helpful for the SP to indicate the requested VPN connection is not a regular site but rather is an end-to-end VPN connection, as specific default device configuration parameters may be applied in case of site-vpn-flavor-e2e (e.g., QoS configuration). In order to establish a connection between Site 1 in SP A and Site 2 in SP B spanning across multiple domains, SP A may request the creation of end-to-end connectivity to SP B. The site-vpn-flavor-e2e will be used to indicate that this is an end-to-end connectivity setup and not a regular customer site.

5.5.2. Attaching a Site to a VPN

Due to the multiple site-vpn flavors, the attachment of a site to an L2VPN is done at the site-network-access (logical access) level through the "vpn-attachment" container. The vpn-attachment container is mandatory. The model provides two ways to attach a site to a VPN:

- By referencing the target VPN directly.
- By referencing a VPN policy for attachments that are more complex.

A choice is implemented to allow the user to choose the flavor that provides the best fit.

5.5.2.1. Referencing a VPN

Referencing a vpn-id provides an easy way to attach a particular logical access to a VPN. This is the best way in the case of a single VPN attachment. When referencing a vpn-id, the site-role setting must be added to express the role of the site in the target VPN service topology.

```
 <?xml version="1.0"?>
 <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
  <vpn-services>
     <vpn-service>
       <vpn-id>VPNA</vpn-id>
       <ce-vlan-preservation>true</ce-vlan-preservation>
       <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
     </vpn-service>
     <vpn-service>
       <vpn-id>VPNB</vpn-id>
       <ce-vlan-preservation>true</ce-vlan-preservation>
       <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
     </vpn-service>
  </vpn-services>
  <sites>
    <site>
     <site-id>SITE1</site-id>
    <locations>
     <location>
      <location-id>L1</location-id>
     </location>
    </locations>
    <management>
     <type>customer-managed</type>
       </management>
        <site-network-accesses>
```

```
            <site-network-access>
             <network-access-id>LA1</network-access-id>
                <service>
                  <svc-bandwidth>
                     <bandwidth>
                       <direction>input-bw</direction>
                        <type>bw-per-cos</type>
                         <cir>450000000</cir>
                         <cbs>20000000</cbs>
                         <eir>1000000000</eir>
                         <ebs>200000000</ebs>
                      </bandwidth>
                    </svc-bandwidth>
                     <carrierscarrier>
                       <signaling-type>bgp</signaling-type>
                     </carrierscarrier>
                     <svc-mtu>1514<svc-mtu>
                   </service>
             <vpn-attachment>
              <vpn-id>VPNA</vpn-id>
              <site-role>spoke-role</site-role>
             </vpn-attachment>
            </site-network-access>
            <site-network-access>
             <network-access-id>LA2</network-access-id>
                 <service>
                   <svc-bandwidth>
                      <bandwidth>
                        <direction>input-bw</direction>
                         <type>bw-per-cos</type>
                          <cir>450000000</cir>
                          <cbs>20000000</cbs>
                          <eir>1000000000</eir>
                          <ebs>200000000</ebs>
                      </bandwidth>
                    </svc-bandwidth>
                     <carrierscarrier>
                       <signaling-type>bgp</signaling-type>
                     </carrierscarrier>
                     <svc-mtu>1514<svc-mtu>
                   </service>
             <vpn-attachment>
              <vpn-id>VPNB</vpn-id>
              <site-role>spoke-role</site-role>
             </vpn-attachment>
            </site-network-access>
           </site-network-accesses>
      </site>
```

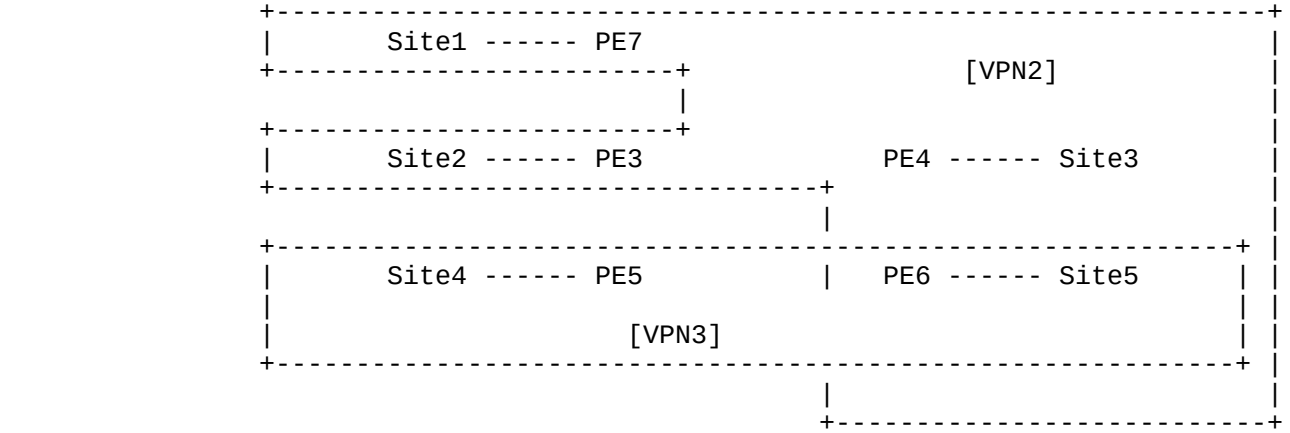
```
  </sites>
 </l2vpn-svc>
```

The example above describes a multi-VPN case where a site (SITE1) has two logical accesses (LA1 and LA2), attached to both VPNA and VPNB.

5.5.2.2. VPN Policy

The "vpn-policy" list helps express a multi-VPN scenario where a logical access belongs to multiple VPNs.

As a site can belong to multiple VPNs, the vpn-policy list may be composed of multiple entries. A filter can be applied to specify that only some LANs at the site should be part of a particular VPN. A site can be composed by multiple LAN segments and each LAN segment can be connected to different VPN. Each time a site (or LAN) is attached to a VPN, the user must precisely describe its role (site-role) within the target VPN service topology.



In the example above, Site5 is part of two VPNs: VPN3 and VPN2. It will play a Hub role in VPN2 and an any-to-any role in VPN3. We can express such a multi-VPN scenario as follows:

```
<?xml version="1.0"?>
 <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
  <vpn-services>
   <vpn-service>
    <vpn-id>VPN2</vpn-id>
    <ce-vlan-preservation>true</ce-vlan-preservation>
    <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
   </vpn-service>
   <vpn-service>
```

```
       <vpn-id>VPN3</vpn-id>
       <ce-vlan-preservation>true</ce-vlan-preservation>
       <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
      </vpn-service>
     </vpn-services>
     <sites>
        <site>
       <locations>
        <location>
         <location-id>L1</location-id>
        </location>
       </locations>
       <management>
        <type>customer-managed</type>
       </management>
          <site-id>Site5</site-id>
          <vpn-policies>
           <vpn-policy>
            <vpn-policy-id>POLICY1</vpn-policy-id>
            <entries>
             <id>ENTRY1</id>
             <vpn>
              <vpn-id>VPN2</vpn-id>
              <site-role>hub-role</site-role>
             </vpn>
            </entries>
            <entries>
             <id>ENTRY2</id>
             <vpn>
              <vpn-id>VPN3</vpn-id>
              <site-role>any-to-any-role</site-role>
             </vpn>
            </entries>
           </vpn-policy>
          </vpn-policies>
          <site-network-accesses>
           <site-network-access>
            <network-access-id>LA1</network-access-id>
         <site>
          <site-id>SITE1</site-id>
       <locations>
        <location>
         <location-id>L1</location-id>
        </location>
       </locations>
       <management>
        <type>customer-managed</type>
       </management>
```

```
          <site-network-accesses>
           <site-network-access>
            <network-access-id>LA1</network-access-id>
                <service>
                  <svc-bandwidth>
                     <bandwidth>
                       <direction>input-bw</direction>
                        <type>bw-per-cos</type>
                         <cir>450000000</cir>
                         <cbs>20000000</cbs>
                         <eir>1000000000</eir>
                         <ebs>200000000</ebs>
                      </bandwidth>
                    </svc-bandwidth>
                     <carrierscarrier>
                       <signaling-type>bgp</signaling-type>
                     </carrierscarrier>
                     <svc-mtu>1514<svc-mtu>
                   </service>
            <vpn-attachment>
             <vpn-id>VPNA</vpn-id>
             <site-role>spoke-role</site-role>
            </vpn-attachment>
           </site-network-access>
           <site-network-access>
            <network-access-id>LA2</network-access-id>
                <service>
                  <svc-bandwidth>
                     <bandwidth>
                       <direction>input-bw</direction>
                        <type>bw-per-cos</type>
                         <cir>450000000</cir>
                         <cbs>20000000</cbs>
                         <eir>1000000000</eir>
                         <ebs>200000000</ebs>
                     </bandwidth>
                    </svc-bandwidth>
                     <carrierscarrier>
                       <signaling-type>bgp</signaling-type>
                     </carrierscarrier>
                     <svc-mtu>1514<svc-mtu>
                   </service>
            <vpn-attachment>
             <vpn-id>VPNB</vpn-id>
             <site-role>spoke-role</site-role>
            </vpn-attachment>
           </site-network-access>
          </site-network-accesses>
```

```
        </site>
           <vpn-attachment>
            <vpn-policy-id>POLICY1</vpn-policy-id>
           </vpn-attachment>
         </site-network-access>
        </site-network-accesses>
       </site>
   </sites>
  </l2vpn-svc>
```

Now, if a more-granular VPN attachment is necessary, filtering can be used. For example, if LAN1 from Site5 must be attached to VPN2 as a Hub and LAN2 must be attached to VPN3, the following configuration can be used:

```
 <?xml version="1.0"?>
   <l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
     <vpn-services>
       <vpn-service>
         <vpn-id>VPN2</vpn-id>
         <ce-vlan-preservation>true</ce-vlan-preservation>
         <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
         </vpn-service>
         <vpn-service>
          <vpn-id>VPN3</vpn-id>
          <ce-vlan-preservation>true</ce-vlan-preservation>
          <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
         </vpn-service>
     </vpn-services>
    <site>
    <locations>
     <location>
      <location-id>L1</location-id>
     </location>
    </locations>
    <management>
     <type>customer-managed</type>
    </management>
          <site-id>Site5</site-id>
          <vpn-policies>
           <vpn-policy>
            <vpn-policy-id>POLICY1</vpn-policy-id>
            <entries>
             <id>ENTRY1</id>
             <filters>
               <filter>
                 <type>lan</type>
                 <lan-tag>LAN1</lan-tag>
```

```
                </filter>
               </filters>
               <vpn>
                <vpn-id>VPN2</vpn-id>
                <site-role>hub-role</site-role>
               </vpn>
              </entries>
              <entries>
               <id>ENTRY2</id>
               <filters>
                 <filter>
                   <type>lan</type>
                   <lan-tag>LAN2</lan-tag>
                 </filter>
               </filters>
                <vpn>
                <vpn-id>VPN3</vpn-id>
                <site-role>any-to-any-role</site-role>
               </vpn>
              </entries>
             </vpn-policy>
            </vpn-policies>
            <site-network-accesses>
             <site-network-access>
              <network-access-id>LA1</network-access-id>
                <service>
                  <svc-bandwidth>
                    <bandwidth>
                      <direction>input-bw</direction>
                      <type>bw-per-cos</type>
                       <cir>450000000</cir>
                       <cbs>20000000</cbs>
                        <eir>1000000000</eir>
                        <ebs>200000000</ebs>
                     </bandwidth>
                    </svc-bandwidth>
                     <carrierscarrier>
                        <signaling-type>bgp</signaling-type>
                     </carrierscarrier>
                      <svc-mtu>1514<svc-mtu>
                 </service>
              <vpn-attachment>
               <vpn-policy-id>POLICY1</vpn-policy-id>
              </vpn-attachment>
             </site-network-access>
            </site-network-accesses>
           </site>
          </sites>
```

```
</l2vpn-svc>
```

5.6. Deciding Where to Connect the Site

The management system will have to determine where to connect each site-network-access of a particular site to the provider network (e.g., PE or aggregation switch).

This model defines parameters and constraints that can influence the meshing of the site-network-access.

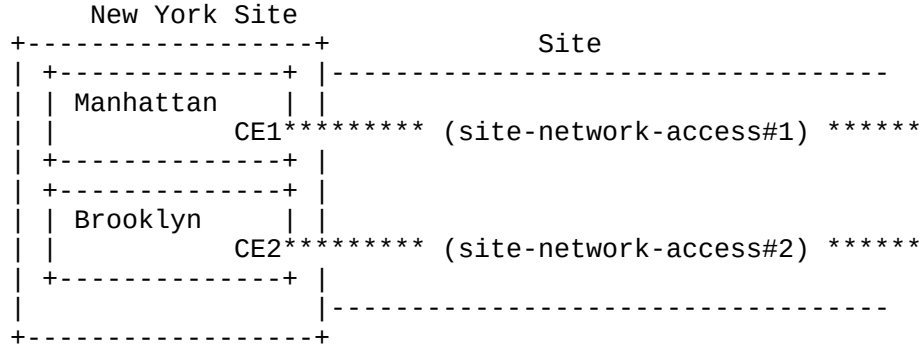
The management system MUST honor all customer constraints, or if a constraint is too strict and cannot be fulfilled, the management system MUST NOT provision the site and MUST provide information to the user about which constraints that could not be fulfilled. How the information is provided is out of scope for this document. Whether or not to relax the constraint would then be left up to the user.

Parameters such as site location (see Section 5.6.2) and access type (see Section 5.6.3) affect the service placement that the management system applies.

In addition to parameters and constraints, the management system's decision MAY be based on any other internal constraints that are left up to the SP, such as least load, distance, etc.

5.6.1. Constraint: Device

In the case of provider management or co-management, one or more devices have been ordered by the customer to a particular location that has already been configured. The customer may force a particular site-network-access to be connected on a particular device that he ordered.

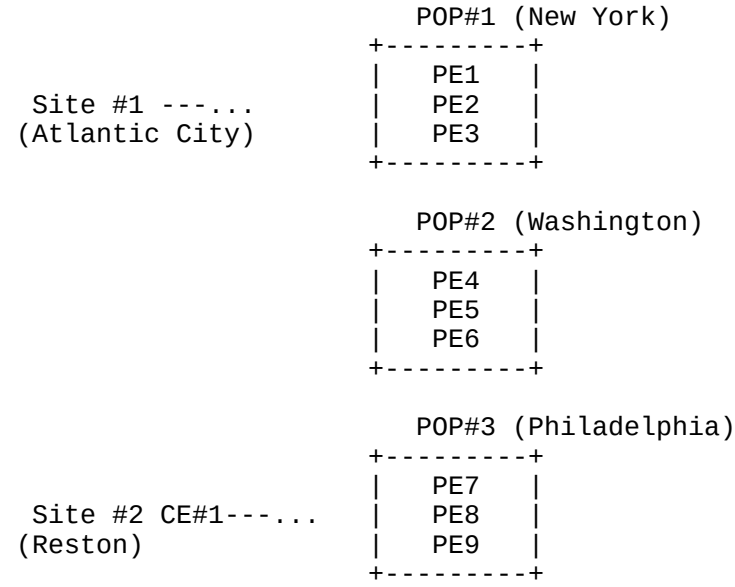


In the figure above, site-network-access#1 is associated with CE1 in the service request. The SP must ensure the provisioning of this connection.

5.6.2. Constraint/Parameter: Site Location

The location information provided in this model MAY be used by a management system to determine the target PE to mesh the site (SP side). A particular location must be associated with each site network access when configuring it. The SP MUST honor the termination of the access on the location associated with the site network access (customer side). The "country-code" in the site location should be expressed as an ISO 3166 code and is similar to country label defined in [RFC4119].

The site-network-access location is determined by the "location-flavor". In the case of a provider-managed or co-managed site, the user is expected to configure a "device-reference" (device case) that will bind the site-network-access to a particular device that the customer ordered. As each device is already associated with a particular location, in such a case the location information is retrieved from the device location. In the case of a customer-managed site, the user is expected to configure a "location-reference" (location case); this provides a reference to an existing configured location and will help with placement.



In the example above, Site #1 is a customer-managed site with a location L1, while Site #2 is a provider-managed site for which a CE (CE#1) was ordered. Site #2 is configured with L2 as its location. When configuring a site-network-access for Site #1, the user will need to reference location L1 so that the management system will know that the access will need to terminate on this location. Then, for distance reasons, this management system may mesh Site #1 on a PE in the Philadelphia POP. It may also take into account resources available on PEs to determine the exact target PE (e.g., least loaded). For Site #2, the user is expected to configure the site-network-access with a device-reference to CE#1 so that the management system will know that the access must terminate on the location of CE#1 and must be connected to CE#1. For placement of the SP side of the access connection, in the case of the nearest PE used, it may mesh Site #2 on the Washington POP.

5.6.3. Constraint/Parameter: Access Type

The management system needs to elect the access media to connect the site to the customer (for example, xDSL, leased line, Ethernet backhaul). The customer may provide some parameters/constraints that will provide hints to the management system.

The bearer container information SHOULD be the first piece of information considered when making this decision:

- The "requested-type" parameter provides information about the media type that the customer would like to use. If the "strict" leaf is equal to "true", this MUST be considered a strict constraint so that the management system cannot connect the site with another media type. If the "strict" leaf is equal to "false" (default) and if the requested media type cannot be fulfilled, the management system can select another media type. The supported media types SHOULD be communicated by the SP to the customer via a mechanism that is out of scope for this document.

- The "always-on" leaf defines a strict constraint: if set to true, the management system MUST elect a media type that is "always-on" (e.g., this means no dial access type).

- The "bearer-reference" parameter is used in cases where the customer has already ordered a network connection to the SP apart from the L2VPN site and wants to reuse this connection. The string used is an internal reference from the SP and describes the already-available connection. This is also a strict requirement that cannot be relaxed. How the reference is given to the customer is out of scope for this document, but as an example, when the customer ordered the bearer (through a process that is

out of scope for this model), the SP may have provided the bearer reference that can be used for provisioning services on top.

Any other internal parameters from the SP can also be used. The management system MAY use other parameters, such as the requested "svc-input-bandwidth" and "svc-output-bandwidth", to help decide which access type to use.

5.6.4. Constraint: Access Diversity

Each site-network-access may have one or more constraints that would drive the placement of the access. By default, the model assumes that there are no constraints, but allocation of a unique bearer per site-network-access is expected.

In order to help with the different placement scenarios, a site-network-access may be tagged using one or multiple group identifiers. The group identifier is a string, so it can accommodate both explicit naming of a group of sites (e.g., "multihomed-set1") and the use of a numbered identifier (e.g., 12345678). The meaning of each group-id is local to each customer administrator, and the management system MUST ensure that different customers can use the same group-ids. One or more group-ids can also be defined at the site level; as a consequence, all site-network-accesses under the site MUST inherit the group-ids of the site they belong to. When, in addition to the site group-ids some group-ids are defined at the site-network-access level, the management system MUST consider the union of all groups (site level and site network access level) for this particular site-network-access.

For an already-configured site-network-access, each constraint MUST be expressed against a targeted set of site-network-accesses. This site-network-access (i.e. the already-configured site-network-access) MUST never be taken into account in the targeted set of site-network-accesses. For example, "My site-network-access S must not be connected on the same POP as the site-network-accesses that are part of Group 10." The set of site-network-accesses against which the constraint is evaluated can be expressed as a list of groups, "all-other-accesses", or "all-other-groups". The all-other-accesses option means that the current site-network-access constraint MUST be evaluated against all the other site-network-accesses belonging to the current site. The all-other-groups option means that the constraint MUST be evaluated against all groups that the current site-network-access does not belong to.

The current model defines multiple constraint-types:

- pe-diverse: The current site-network-access MUST NOT be connected to the same PE as the targeted site-network-accesses.

- pop-diverse: The current site-network-access MUST NOT be connected to the same POP as the targeted site-network-accesses.

- linecard-diverse: The current site-network-access MUST NOT be connected to the same linecard as the targeted site-network-accesses. Note that customer can request lincard-diverse for site-network-accesses, but the specific linecard identifier used should not be exposed to customer.

- bearer-diverse: The current site-network-access MUST NOT use common bearer components compared to bearers used by the targeted site-network-accesses. "bearer-diverse" provides some level of diversity at the access level. As an example, two bearer-diverse site-network-accesses must not use the same DSLAM, BAS, or Layer 2 switch.

- same-pe: The current site-network-access MUST be connected to the same PE as the targeted site-network-accesses.

- same-bearer: The current site-network-access MUST be connected using the same bearer as the targeted site-network-accesses.

These constraint-types can be extended through augmentation. Each constraint is expressed as "The site-network-access S must be <constraint-type> (e.g., pe-diverse, pop-diverse) from these <target> site-network-accesses."

The group-id used to target some site-network-accesses may be the same as the one used by the current site-network-access. This eases the configuration of scenarios where a group of site-network-access points has a constraint between the access points in the group.

5.7. Route Distinguisher and Network Instance Allocation

The route distinguisher (RD) is a critical parameter of BGP-based L2VPNs as described in [RFC4364] that provides the ability to distinguish common addressing plans in different VPNs. As for route targets (RTs), a management system is expected to allocate a MAC-VRF on the target PE and an RD for this MAC-VRF. This RD MUST be unique across all MAC-VRFs on the target PE.

If a MAC-VRF already exists on the target PE and the MAC-VRF fulfills the connectivity constraints for the site, there is no need to recreate another MAC-VRF, and the site MAY be meshed within this existing MAC-VRF. How the management system checks that an existing

MAC-VRF fulfills the connectivity constraints for a site is out of scope for this document.

If no such MAC-VRF exists on the target PE, the management system has to initiate the creation of a new MAC-VRF on the target PE and has to allocate a new RD for this new MAC-VRF.

The management system MAY apply a per-VPN or per-MAC-VRF allocation policy for the RD, depending on the SP's policy. In a per-VPN allocation policy, all MAC-VRFs (dispatched on multiple PEs) within a VPN will share the same RD value. In a per-MAC-VRF model, all MAC-VRF should always have a unique RD value. Some other allocation policies are also possible, and this document does not restrict the allocation policies to be used.

The allocation of RDs MAY be done in the same way as RTs. The examples provided in Section 5.2.2.1 could be reused in this scenario.

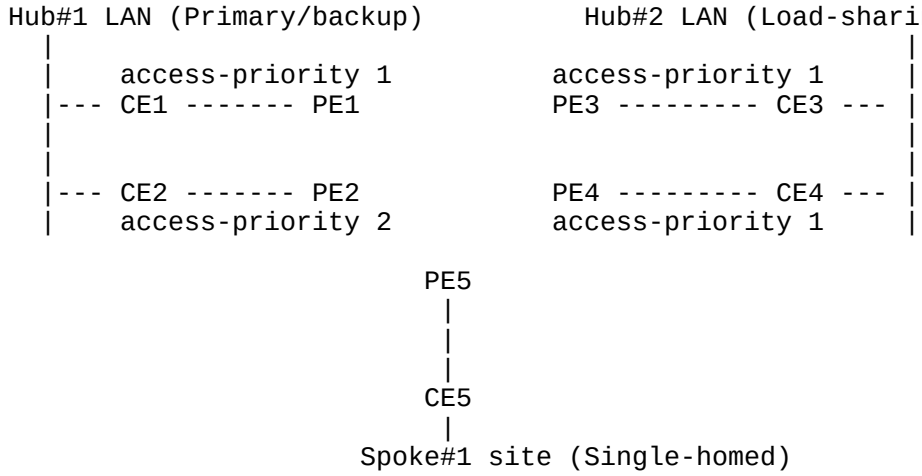
Note that an SP MAY configure a target PE for an automated allocation of RDs. In this case, there will be no need for any backend system to allocate an RD value.

5.8. Site Network Access Availability

A site may be multihomed, meaning that it has multiple site-network-access points. Placement constraints defined in previous sections will help ensure physical diversity.

When the site-network-accesses are placed on the network, a customer may want to use a particular routing policy on those accesses. The "site-network-access/availability" container defines parameters for site redundancy. The "access-priority" leaf defines a preference for a particular access. This preference is used to model load-balancing or primary/backup scenarios. The higher the access-priority value, the higher the preference will be. The "redundancy mode" attribute is defined for an multi-homing site and used to model single-active and active/active scenarios. It allows for multiple active paths in forwarding state and for load-balancing options.

The figure below describes how the access-priority attribute can be used.



In the figure above, Hub#2 requires load-sharing, so all the site-network-accesses must use the same access-priority value. On the other hand, as Hub#1 requires a primary site-network-access and a backup site-network-access, a higher access-priority setting will be configured on the primary site-network-access.

Scenarios that are more complex can also be modeled. Let’s consider a Hub site with five accesses to the network (A1, A2, A3, A4, and A5). The customer wants to load-share its traffic on A1 and A2 in the nominal situation. If A1 and A2 fail, the customer wants to load-share its traffic on A3 and A4; finally, if all of A1 to A4 are down, he wants to use A5. We can model this easily by configuring the following access-priority values: A1=100, A2=100, A3=50, A4=50, A5=10.

The access-priority scenario has some limitations. An access-priority scenario like the previous one with five accesses but with the constraint of having traffic load-shared between A3 and A4 in the case where just one of A1 or A2 is down is not achievable. But the access-priority attribute defined will cover most of the deployment use cases, and if necessary the model can be extended via augmentation to support additional use cases.

5.9. SVC MTU

The maximum MTU of subscriber service frames can be derived from the physical interface MTU by default, or specified under the "svc-mtu" leaf if it is different than the default number.

5.10. Service

The "service" container defines service parameters associated with the site.

5.10.1. Bandwidth

The service bandwidth refers to the bandwidth requirement between CE and PE and can be represented using Committed Information Rate(CIR),Excess Information Rate(EIR),Peak Information Rate(PIR). The requested bandwidth is expressed as ingress bandwidth and egress bandwidth. Ingress/egress direction uses the customer site as the point of reference: Ingress direction bandwidth means download bandwidth for the site, and egresss bandwidth means upload bandwidth for the site.

The service bandwidth is only configurable at the site-network-access level (i.e., for the site network access associated with the site).

Using a different ingress and egress bandwidth will allow service provider to know if a customer allows for asymmetric bandwidth access like ADSL. It can also be used to set the rate limit in a different way for upload and download on symmetric bandwidth access.

The svc-bandwidth has specific type. This document defines four types:

- bw-per-access Bandwidth is per connection or site network access, providing rate enforcement for all service frames at the interface that are associated with a particular network access.
- bw-per-cos Bandwidth is per cos, providing rate enforcement for all service frames for a given class of service with specific cos-id.
- bw-per-svc bandwidth is per site, providing rate enforcement for all service frames that are associated with a particular VPN service.
- opaque bandwidth is the total bandwidth that is not associated with any particular cos-id, vpn service identified with the vpn-id, or site network access id.

The svc-bandwidth must include a "cos-id" parameter if the ’type’ is set as ’bw-per-cos’. The cos-id can be assigned based on the IEEE 802.1p value in the C-tag, or on the DSCP in the Ethernet Frame header. Service frames are metered against the bandwidth profile based on the cos-identifier.

The svc-bandwidth must be associated with a specific "site-network-access-id" parameter if the 'type' is set as 'bw-per-access'. Multiple bandwidths per cos-id can be associated with the same Site Network access.

The svc-bandwidth must include a specific "vpn-id" parameter if the 'type' is set as 'bw-per-svc'. Multiple bandwidths per cos-id can be associated with the same Ethernet VPN service.

5.10.2. QoS

The model defines QoS parameters as an abstraction:

- qos-classification-policy: Defines a set of ordered rules to classify customer traffic.

- qos-profile: Provides a QoS scheduling profile to be applied.

5.10.2.1. QoS Classification

QoS classification rules are handled by qos-classification-policy. The qos-classification-policy is an ordered list of rules that match a flow or application and set the appropriate target class of service (target-class-id). The user can define the match using a more specific flow definition (based on layer 2 source and destination MAC addresses, cos, dscp, cos-id, color-id, etc.). A "color-id" will be assigned to a service frame to identify its QoS profile conformance. A service frame is "green" if it is conformant with the "committed" rate of the bandwidth profile. A Service Frame is "yellow" if it is exceeding the "committed" rate, but conformant with the "excess" rate of the bandwidth profile. Finally, a service frame is "red" if it is conformant with neither the "committed" nor "excess" rates of the bandwidth profile.

When a flow definition is used, the user can use a target-sites leaf-list to identify the destination of a flow rather than using destination addresses. In such a case, an association between the site abstraction and the MAC addresses used by this site must be done dynamically. How this association is done is out of scope for this document. The association of a site to an L2VPN is done through the "vpn-attachment" container. Therefore, the user can also employ the "target-sites" leaf-list and "vpn-attachment" to identify the destination of a flow targeted to specific VPN service. A rule that does not have a match statement is considered as a match-all rule. A service provider may implement a default terminal classification rule if the customer does not provide it. It will be up to the service provider to determine its default target class. This model defines some applications, but new application identities may be added

through augmentation. The exact meaning of each application identity is up to the SP, so it will be necessary for the SP to advise the customer on the usage of application matching.

5.10.2.2. QoS Profile

A user can choose between the standard profile provided by the operator or a custom profile. The qos-profile defines the traffic scheduling policy to be used by the service provider.

A custom qos-profile is defined as a list of class of services and associated properties. The properties are:

- direction: Used to specify the direction to which the qos-profile is applied. This model supports "Site-to-WAN" direction, "WAN-to-Site" direction, and "both" directions. By default, "both" directions is used. In case of "both" directions, the provider should ensure scheduling according to the requested policy in both traffic directions (SP to customer, and customer to SP). As an example, a device-scheduling policy may be implemented on both the PE side and the CE side of the WAN link. In case of "WAN-to-Site" direction, the provider should ensure scheduling from the SP network to the customer site. As an example, a device- scheduling policy may be implemented only on the PE side of the WAN link towards the customer.

- policing: The optional "policing" indicates whether policing should apply to one-rate two-colors or to two-rates three-colors.

- byte-offset: The optional "byte-offset" indicates how many bytes in the service frame header are excluded from rate enforcement.

- frame-delay: Used to define the latency constraint of the class. The latency constraint can be expressed as the lowest possible latency, or as a latency boundary expressed in milliseconds. How this latency constraint will be fulfilled is up to the service provider implementation: a strict priority queueing mechanism may be used on the access and in the core network, or a low latency routing path may be created for this traffic class.

- frame-jitter: Used to define the jitter constraint of the class. The jitter constraint can be expressed as the lowest possible jitter, or as a jitter boundary expressed in microseconds. How this jitter constraint will be fulfilled is up to the service provider implementation: a strict priority queueing mechanism may be used on the access and in the core network, or a jitter-aware routing path may be created for this traffic class.

- bandwidth: used to define a guaranteed amount of bandwidth for the class of service. It is expressed as a percentage. The "guaranteed-bw-percent" parameter uses available bandwidth as a reference. The available bandwidth should not fall below the Committed Information Rate (CIR) defined under svc-input-bandwidth or svc-output-bandwidth. When the qos-profile container is implemented on the CE side, svc-output-bandwidth is taken into account as a reference. When it is implemented on the PE side, svc-input-bandwidth is used. By default, the bandwidth reservation is only guaranteed at the access level. The user can use the "end-to-end" leaf to request an end-to-end bandwidth reservation, including across the MPLS transport network. (In other words, the SP will activate something in the MPLS core to ensure that the bandwidth request from the customer will be fulfilled by the MPLS core as well.) How this is done (e.g., RSVP-TE reservation, controller reservation) is out of scope for this document.

In addition, due to network conditions, some constraints may not be completely fulfilled by the SP; in this case, the SP should advise the customer about the limitations. How this communication is done is out of scope for this document.

5.10.3. Broadcast Multicast Unknow Unicast Support

The "broadcast-unknown-unicast-multicast" container defines the type of site in the customer multicast service topology: source, receiver, or both. These parameters will help the management system optimize the multicast service.

Multiple multicast group-to-port mappings can be created using the "multicast-gp-address-mapping" list. The "multicast-gp-address-mapping" defines multicast group address and port LAG number. Those parameters will help the SP select the appropriate association between interface and multicast group to fulfill the customer service requirement.

A whole Layer-2 multicast frame (whether for data or control) should not be altered from a CE to CEs except for the VLAN ID field, ensuring that it is transparently transported. If VLAN IDs are assigned by the SP, they can also be altered.

For point-to-point services, the provider only needs to deliver a single copy of each service frame to the remote PE, regardless whether the destination MAC address of the incoming frame is unicast, multicast or broadcast. Therefore, all service frames should be delivered unconditionally.

BUM (Broadcast-UnknownUnicast-Multicast) frame forwarding in multipoint-to-multipoint services, on the other hand, involves both local flooding to other attachment circuits on the same PE and remote replication to all other PEs, thus consumes additional resources and core bandwidth. Special BUM frame disposition rules can be implemented at external facing interfaces (UNI or E-NNI) to rate-limit the BUM frames, in term of number of packets per second or bits per second.

The threshold can apply to all BUM traffic, or one for each category.

5.11. Site Management

The "management" sub-container is intended for site management options, depending on the device ownership and security access control. The followings are three common management models:

CE Provider Managed: The provider has the sole ownership of the CE device. Only the provider has access to the CE. The responsibility boundary between SP and customer is between CE and customer network. This is the most common use case.

CE Customer Managed: The customer has the sole ownership of the CE device. Only the customer has access to the CE. In this model, the responsibility boundary between SP and customer is between PE and CE.

CE Co-managed: The provider has ownership of the CE device and responsible for managing the CE. However, the provider grants the customer access to the CE for some configuration/monitoring purposes. In this co-managed mode, the responsibility boundary is the same as for the provider-managed model.

The selected management mode is specified under the "type" leaf. The "address" leaf stores CE device management addressing information. And the "management-transport" leaf is used to identify the transport protocol for management traffic: IPv4 or IPv6. Additional security options may be derived based on the particular management model selected.

5.12. MAC Loop Protection

MAC address flapping between different physical ports typically indicates a bridge loop condition in the customer network. Misleading entries in the MAC cache table can cause service frames to circulate around the network indefinitely and saturate the links throughout the provider’s network, affecting other services in the

same network. In case of EVPN, it also introduces massive BGP updates and control plane instability.

The service provider may opt to implement a switching loop prevention mechanism at the external facing interfaces for multipoint-to-multipoint services by imposing a MAC address move threshold.

The MAC move rate and prevention-type options are listed in the "mac-loop-prevention" container.

5.13. MAC Address Limit

The optional "mac-address-limit" container contains the customer MAC address limit and information to describe the action when the limit is exceeded and the aging time for a MAC address.

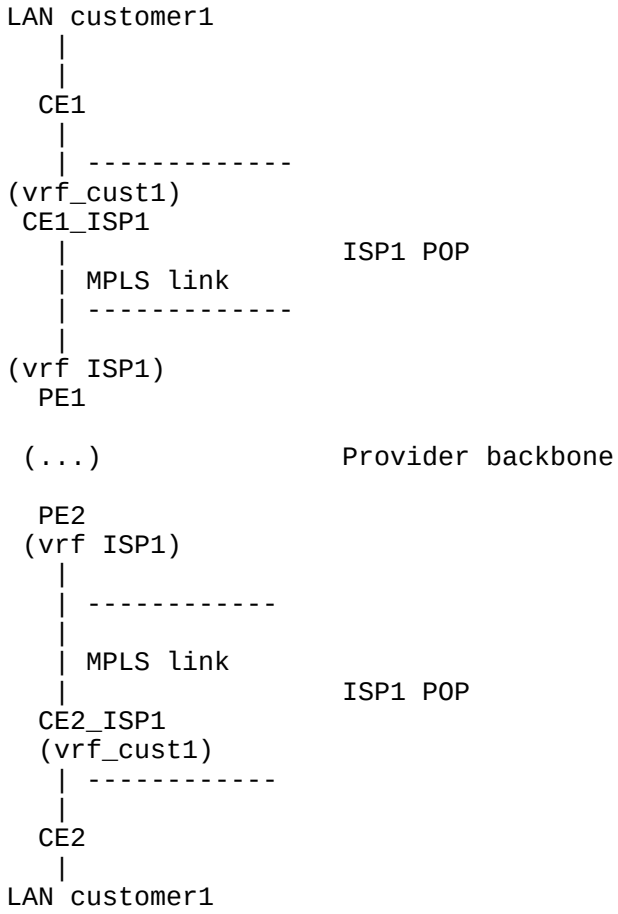
When multiple services are provided on the same network element, the MAC address table (and the Routing Information Base space for MAC-routes in the case of EVPN) is a shared common resource. Service providers may impose a maximum number of MAC addresses learned from the customer for a single service instance by using 'mac-limit'leaf, and may use 'action' leaf to specify the action when the upper limit is exceeded: drop the packet, flood the packet, or simply send a warning log message.

For point-to-point services, if MAC learning is disabled then the MAC address limit is not necessary.

5.14. Enhanced VPN Features

5.14.1. Carriers' Carriers

In the case of Carriers' Carrier (CsC) [RFC6624], a customer may want to build an MPLS service using an L2VPN to carry its traffic.



In the figure above, ISP1 resells an L2VPN service but has no core network infrastructure between its POPs. ISP1 uses an L2VPN as the core network infrastructure (belonging to another provider) between its POPs.

In order to support CsC, the VPN service must indicate MPLS support by setting the "carrierscarrier" leaf to true in the vpn-service list. The link between CE1_ISP1/PE1 and CE2_ISP1/PE2 must also run an MPLS signalling protocol. This configuration is done at the site level.

In this model, LDP or BGP can be used as the MPLS signalling protocol. In the case of LDP, an IGP routing protocol MUST also be activated. In the case of BGP signalling, BGP MUST also be configured as the routing protocol.

If CsC is enabled, the requested "svc-mtu" leaf will refer to the MPLS MTU and not to the link MTU.

5.15. External ID References

The service model sometimes refers to external information through identifiers. As an example, to order cloud-access to a particular cloud service provider (CSP), the model uses an identifier to refer to the targeted CSP. If a customer is directly using this service model as an API (through RESTCONF or NETCONF, for example) to order a particular service, the SP should provide a list of authorized identifiers. In the case of cloud-access, the SP will provide the associated identifiers for each available CSP. The same applies to other identifiers, such as std-qos-profile.

As an usage example, the remote-carrier-name is used in the NNI case because it should be known by the current L2VPN Service Provider it is connecting. While cloud-identifier should be known by both the current L2VPN Service Provider and the customer because it is applied to public cloud or internet access.

How an SP provides the meanings of those identifiers to the customer is out of scope for this document.

5.16. Defining NNIs and Inter-AS support

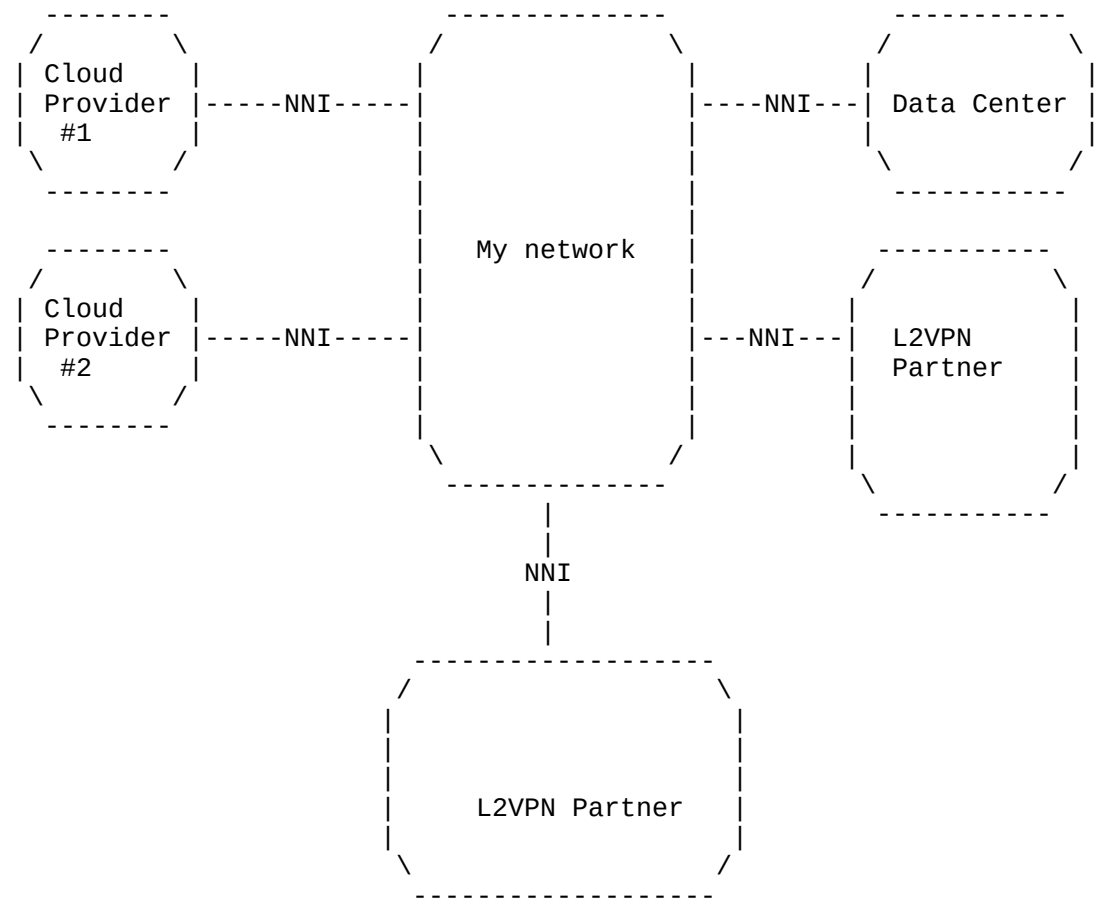
An autonomous system (AS) is a single network or group of networks that is controlled by a common system administration group and that uses a single, clearly defined routing protocol. In some cases, VPNs need to span different ASes in different geographic areas or span different SPs. The connection between ASes is established by the SPs and is seamless to the customer. Examples include:

- A partnership between SPs (e.g., carrier, cloud) to extend their VPN services seamlessly.

- An internal administrative boundary within a single SP (e.g., backhaul versus core versus data center).

NNIs have to be defined to extend the VPNs across multiple ASes. [RFC4761] defines multiple flavors of VPN NNI implementation. Each implementation has pros and cons; this topic is outside the scope of this document. For example, in an Inter-AS option A, autonomous system border router (ASBR) peers are connected by multiple interfaces with at least one of those interfaces spanning the two ASes while being present in the same VPN. In order for these ASBRs to signal label blocks, they associate each interface with a Virtual Switching (MAC-VRF) instance and a BGP session. As a result, traffic

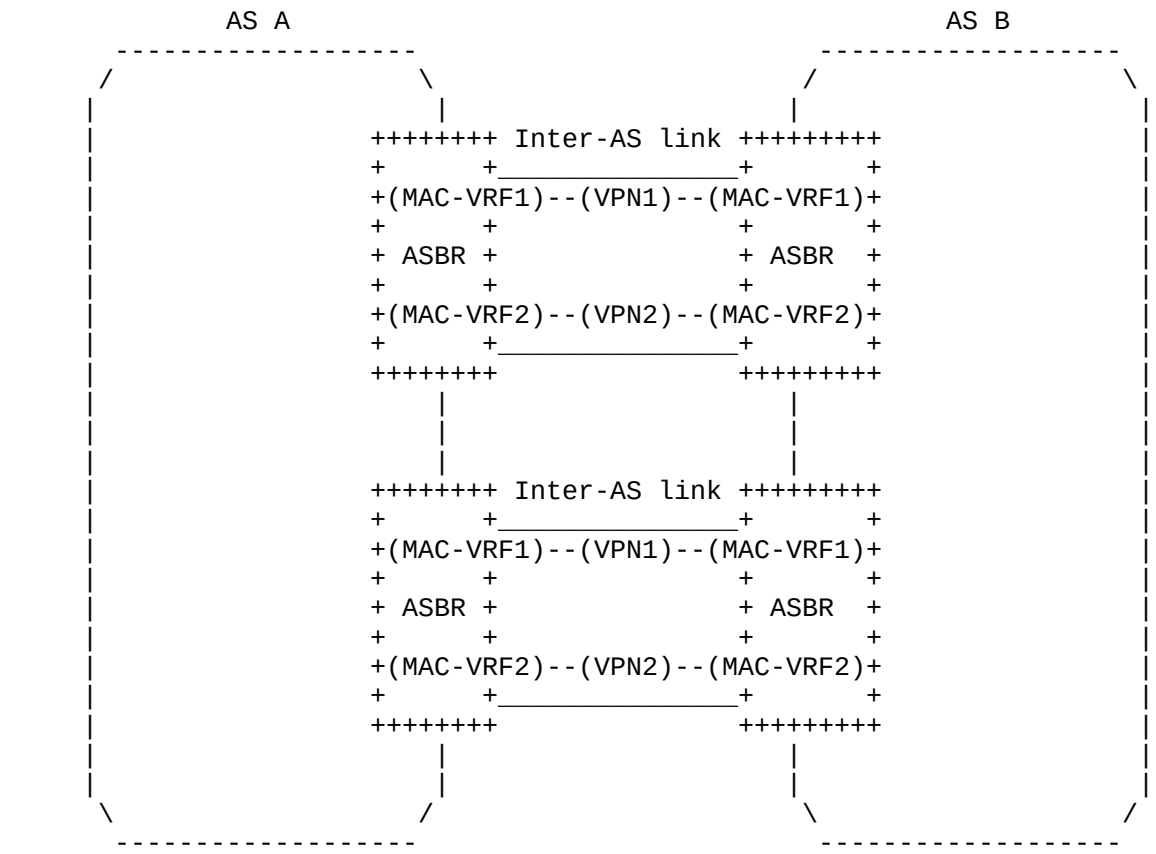
between the back-to-back VPLS is Ethernet. In this scenario, the VPNs are isolated from each other, and because the traffic is Ethernet, QoS mechanisms that operate on Ethernet traffic can be applied to achieve customer service level agreements (SLAs).



The figure above describes an SP network called "My network" that has several NNIs. This network uses NNIs to:

- increase its footprint by relying on L2VPN partners.
- connect its own data center services to the customer L2VPN.
- enable the customer to access its private resources located in a private cloud owned by some CSPs.

5.16.1. Defining an NNI with the Option A Flavor

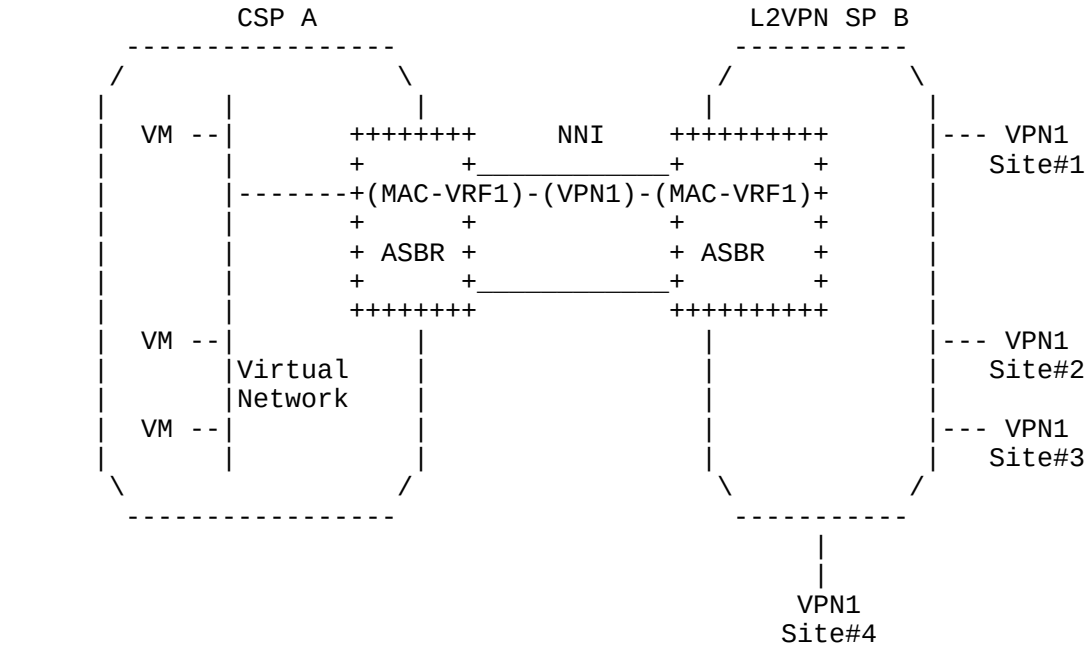


In option A, the two ASes are connected to each other with physical links on ASBRs. For resiliency purposes, there may be multiple physical connections between the ASes. A VPN connection -- physical or logical (on top of physical) -- is created for each VPN that needs to cross the AS boundary, thus providing a back-to-back VPLS model.

From a service model's perspective, this VPN connection can be seen as a site. Let's say that AS B wants to extend some VPN connections for VPN C on AS A. The administrator of AS B can use this service model to order a site on AS A. All connection scenarios could be realized using the features of the current model. As an example, the figure above shows two physical connections that have logical connections per VPN overlaid on them. This could be seen as a multi-VPN scenario. Also, the administrator of AS B will be able to choose the appropriate routing protocol (e.g., E-BGP) to dynamically exchange routes between ASes.

This document assumes that the option A NNI flavor SHOULD re-use the existing VPN site modeling.

Example: a customer wants its CSP A to attach its virtual network N to an existing L2VPN (VPN1) that he has from L2VPN SP B.



To create the VPN connectivity, the CSP or the customer may use the L2VPN service model that SP B exposes. We could consider that, as the NNI is shared, the physical connection (bearer) between CSP A and SP B already exists. CSP A may request through a service model the creation of a new site with a single site-network-access (single-homing is used in the figure). As a placement constraint, CSP A may use the existing bearer reference it has from SP A to force the placement of the VPN NNI on the existing link. The XML below illustrates a possible configuration request to SP B:

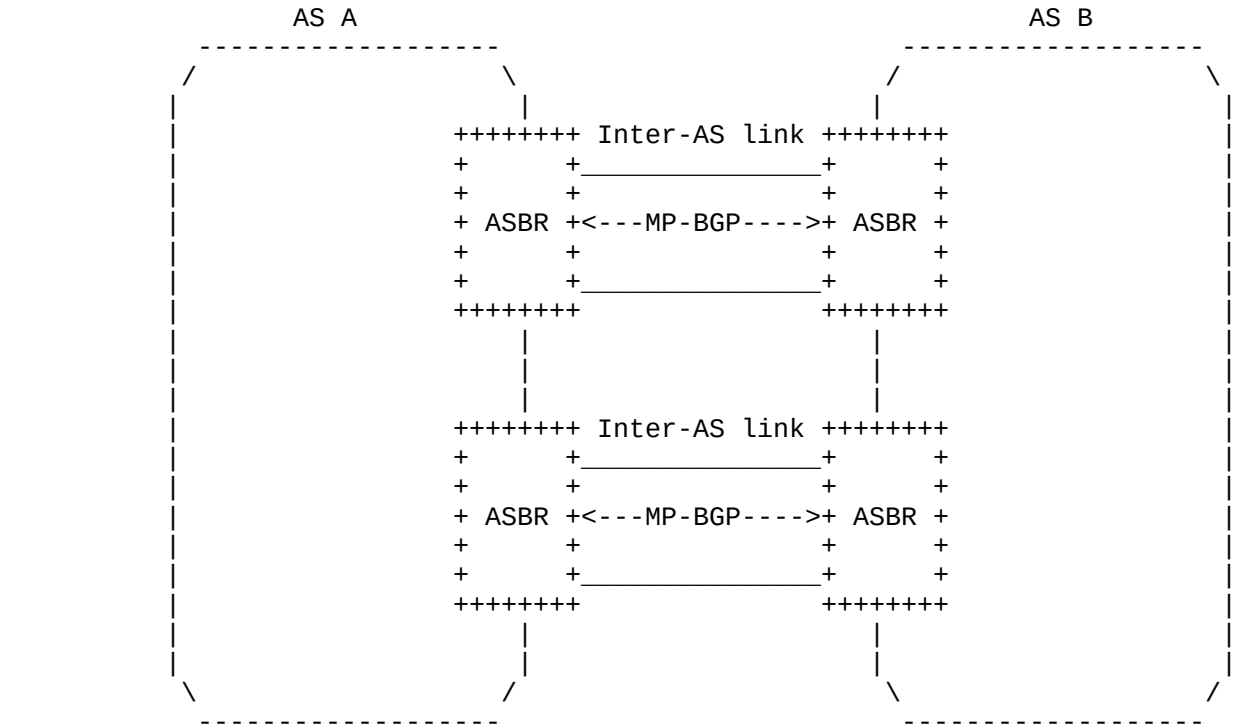
```
 <?xml version="1.0"?>
<l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
 <vpn-profiles>
  <valid-provider-identifiers>
   <qos-profile-identifier>
    <id>GOLD</id>
   </qos-profile-identifier>
   <qos-profile-identifier>
    <id>PLATINUM</id>
```

```
      </qos-profile-identifier>
     </valid-provider-identifiers>
    </vpn-profiles>
    <vpn-services>
     <vpn-service>
      <vpn-id>VPN1</vpn-id>
      <ce-vlan-preservation>true</ce-vlan-preservation>
      <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
     </vpn-service>
    </vpn-services>
    <sites>
         <site>
             <site-id>CSP_A_attachment</site-id>
              <locations>
               <location>
                 <location-id>NY1</location-id>
                 <city>NY</city>
                 <country-code>US</country-code>
              </location>
              </locations>
             <site-vpn-flavor>site-vpn-flavor-nni</site-vpn-flavor>
               <site-network-accesses>
                 <site-network-access>
                  <network-access-id>CSP_A_VN1</network-access-id>
                          <connection>
                          <encapsulation-type>vlan</encapsulation-type>
                          <eth-inf-type>tagged</eth-inf-type>
                           <tagged-interface>
                            <tagged-inf-type>dot1q </tagged-inf-type>
                            <dot1q-vlan-tagged>
                             <cvlan-id>17</cvlan-id>
                           </dot1q-vlan-tagged>
                           </tagged-interface>
                          </connection>
                            <service>
                              <svc-bandwidth>
                                <bandwidth>
                                <direction>input-bw</direction>
                                <type>bw-per-cos</type>
                                <cir>450000000</cir>
                                <cbs>20000000</cbs>
                                <eir>1000000000</eir>
                                <ebs>200000000</ebs>
                                </bandwidth>
                              </svc-bandwidth>
                              <carrierscarrier>
                                <signaling-type>bgp</signaling-type>
                             </carrierscarrier>
```

```
                          </service>
                          <vpn-attachment>
                             <vpn-id>12456487</vpn-id>
                             <site-role>spoke-role</site-role>
                          </vpn-attachment>
                       </site-network-access>
                    </site-network-accesses>
                    <management>
                     <type>customer-managed</type>
                    </management>
                 </site>
              </sites>
             </l2vpn-svc>
```

The case described above is different from a scenario using the cloud-accesses container, as the cloud-access provides a public cloud access while this example enables access to private resources located in a CSP network.

5.16.2. Defining an NNI with the Option B Flavor



In option B, the two ASes are connected to each other with physical links on ASBRs. For resiliency purposes, there may be multiple physical connections between the ASes. The VPN "connection" between ASes is done by exchanging VPN routes through MP-BGP [RFC4761].

There are multiple flavors of implementations of such an NNI. For example:

1. The NNI is internal to the provider and is situated between a backbone and a data center. There is enough trust between the domains to not filter the VPN routes. So, all the VPN routes are exchanged. RT filtering may be implemented to save some unnecessary route states.

2. The NNI is used between providers that agreed to exchange VPN routes for specific RTs only. Each provider is authorized to use the RT values from the other provider.

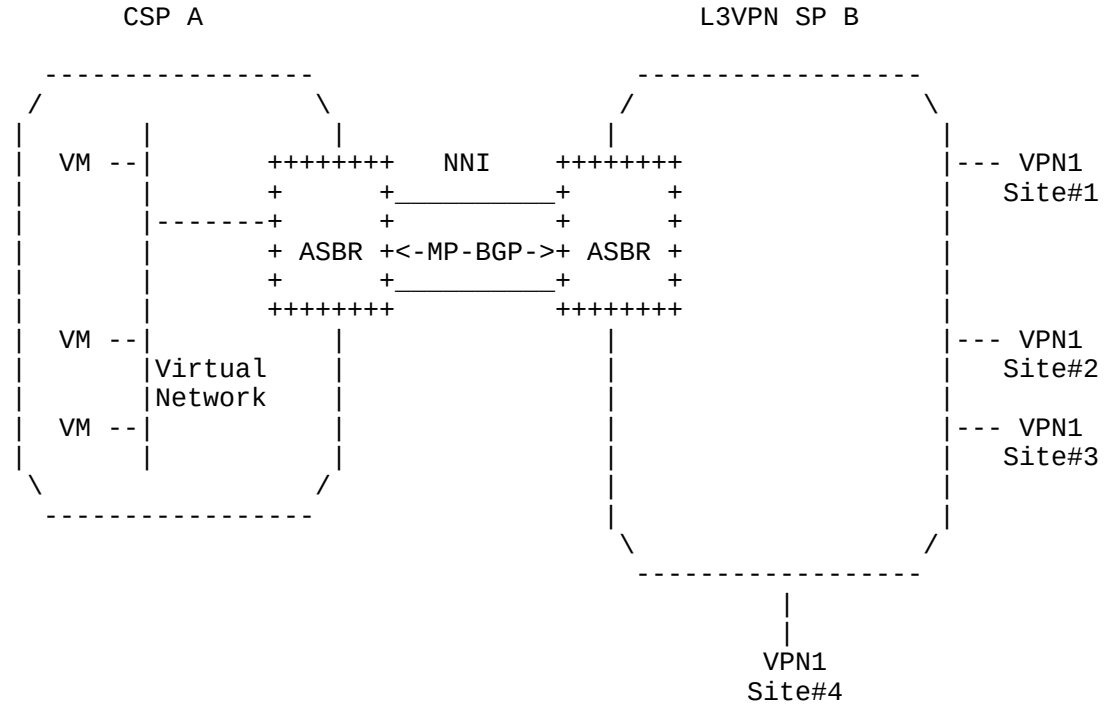
3. The NNI is used between providers that agreed to exchange VPN routes for specific RTs only. Each provider has its own RT scheme. So, a customer spanning the two networks will have different RTs in each network for a particular VPN.

Case 1 does not require any service modeling, as the protocol enables the dynamic exchange of necessary VPN routes.

Case 2 requires that an RT-filtering policy on ASBRs be maintained. From a service modeling point of view, it is necessary to agree on the list of RTs to authorize.

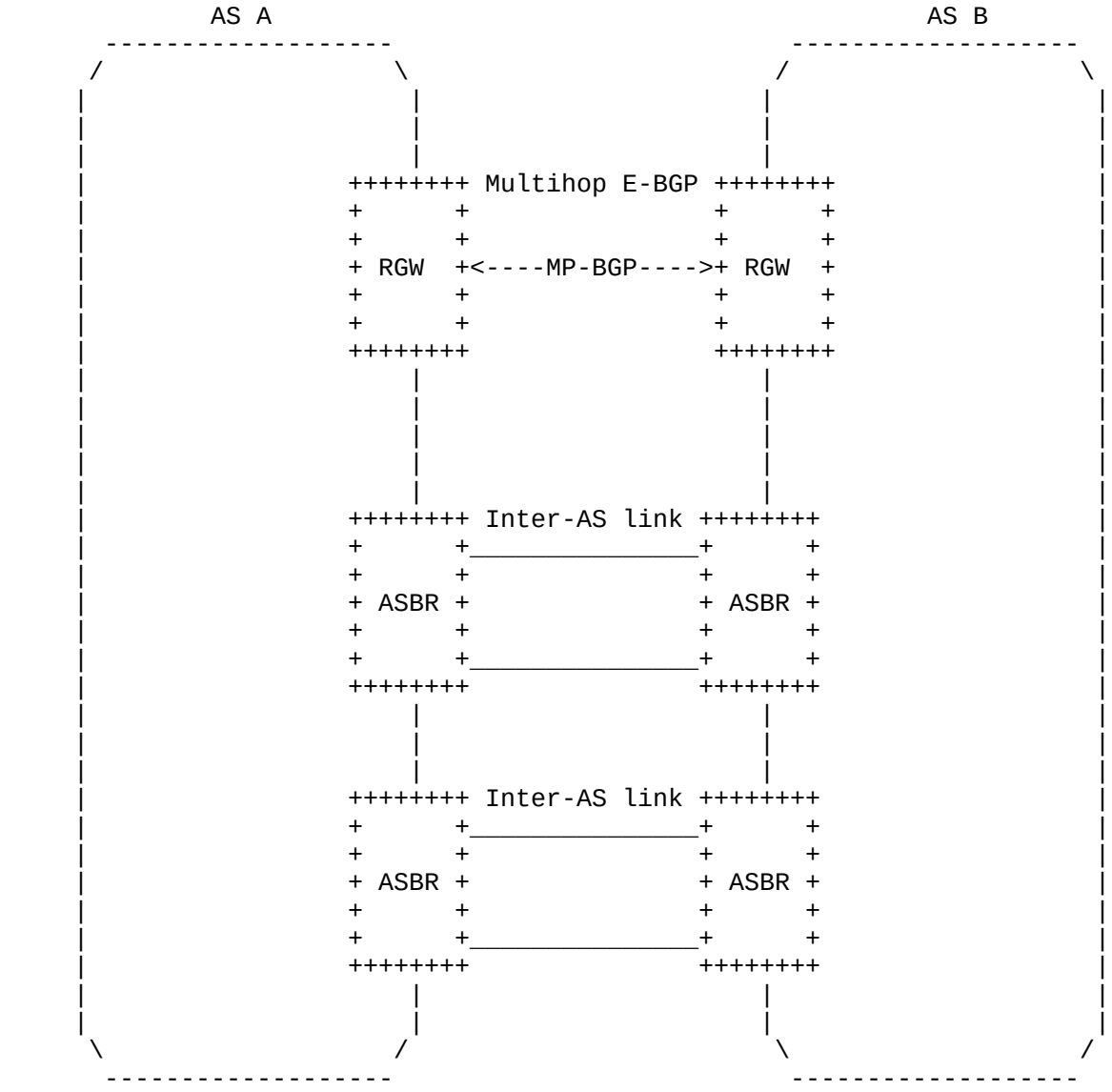
In Case 3, both ASes need to agree on the VPN RT to exchange, as well as how to map a VPN RT from AS A to the corresponding RT in AS B (and vice versa).

Those modelings are currently out of scope for this document.



The example above describes an NNI connection between CSP A and SP network B. The SPs do not trust each other and use different RT allocation policies. So, in terms of implementation, the customer VPN has a different RT in each network (RT A in CSP A and RT B in SP network B). In order to connect the customer's virtual network in CSP A to the customer's L2VPN (VPN1) in SP network B, CSP A should request that SP network B open the customer VPN on the NNI (accept the appropriate RT). Who does the RT translation depends on the agreement between the two SPs: SP B may permit CSP A to request VPN (RT) translation.

5.16.3. Defining an NNI with the Option C Flavor



From a VPN service's perspective, the option C NNI is very similar to option B, as an MP-BGP session is used to exchange VPN routes between the ASes. The difference is that the forwarding plane and the control plane are on different nodes, so the MP-BGP session is multihop between routing gateway (RGW) nodes. From a VPN service's point of view, modeling options B and C will be identical.

5.17. Applicability of L2SM model in Inter-Provider and Inter-Domain Orchestration

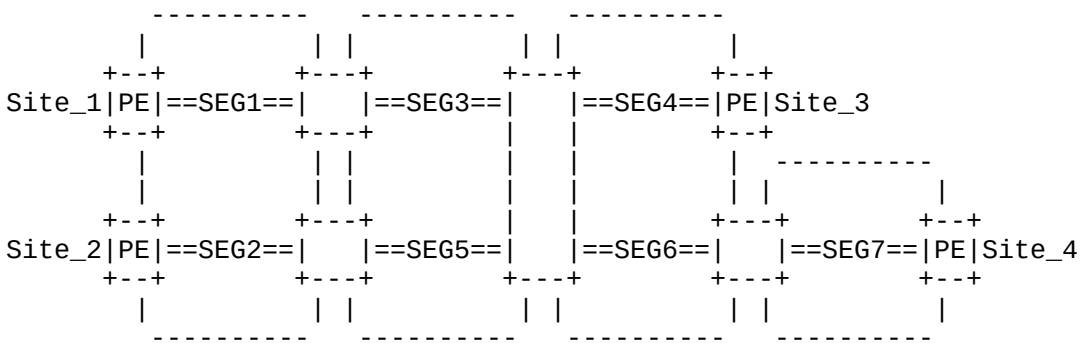
In the case where the ASes belong to different providers, one might imagine that providers would like to have fewer signaling sessions crossing the AS boundary and that the entities that terminate the sessions could be restricted to a smaller set of devices. Two approaches can be taken:

a. Inter-provider control connections to run only between the two border routers

b. Allow an end-to-end, multi-segment connectivity to be constructed out of several connectivity segments, without maintaining an end-to-end control connection.

Inter-provider control connection described in (a) can be realized using techniques of Section 5.15 (i.e., defining NNI). Multi-segment connectivity described in (b) can produce an inter-AS solution that more closely resembles option (b) in [RFC4364]. It may be realized using stitching of per-site connectivity and service segments at different domains, e.g., end-to-end connectivity between Site_1 and Site_3 spans multiple domains (e.g., Metro networks) and can be constructed by stitching network access connectivity within Site_1 with SEG1, SEG3, SEG4 and network access connectivity within Site_3 (as shown in the following figure). The assumption is that service the orchestration component in Figure 3 should have visibility of the complete abstract topology and resource availability. This may rely on network planning.

Note that SEG1,SEG2,SEG3,SEG4,SEG5,SEG6 can also be regarded as network access connectivity within a site and can be created as a normal site using L2SM service model.



In this figure, we use BGP redistribution of L2VPN NLRIs from AS to neighboring AS. First, the PE routers use BGP to redistribute L2VPN NLRIs either to an ASBR, or to a route reflector of which an ASBR is a client. The ASBR then uses BGP to redistribute those L2VPN NLRI to an ASBR in another AS, which in turn distributes them to the PE routers in that AS, or perhaps to another ASBR which in turn distributes them, and so on.

In this case, a PE can learn the address of an ASBR through which it could reach another PE to which it wishes to establish a connectivity. That is, a local PE will receive a BGP advertisement containing L2VPN NLRI corresponding to an L2VPN instance in which the local PE has some attached members. The BGP next-hop for that L2VPN NLRI will be an ASBR of the local AS. Then, rather than building a control connection all the way to the remote PE, it builds one only to the ASBR. A connectivity segment can now be established from the PE to the ASBR. The ASBR in turn can establish a connectivity to the ASBR of the next AS, and stitching that connectivity to the connectivity from the PE as described in [RFC6073]. Repeating the process at each ASBR leads to a sequence of connectivity segments that, when stitching together, connect the two PEs.

Note that in the approach just described, the local PE may never learn the IP address of the remote PE. It learns the L2VPN NLRI advertised by the remote PE, which need not contain the remote PE address, and it learns the IP address of the ASBR that is the BGP next hop for that NLRI.

When this approach is used for VPLS, or for full-mesh VPWS, it leads to a full mesh of connectivity among the PEs, but it does not require a full mesh of control connections (LDP or L2TPv3 sessions). Instead, the control connections within a single AS run among all the PEs of that AS and the ASBRs of the AS. A single control connection between the ASBRs of adjacent ASes can be used to support however many AS-to-AS connectivity segments are needed.

6. Interaction with Other YANG Modules

As expressed in Section 4, this service module is not intended to configure the network element, but is instantiated in a management system.

The management system might follow modular design and comprise at least two different components:

a. The component instantiating the service model (let’s call it the service component)

b. The component responsible for network element configuration (let's call it the configuration component)

In some cases, when a split is needed between the behavior and functions that a customer requests and the technology that the network operator has available to deliver the service [RFC8309], a new component can be separated out of the service component (let's call it the control component). This component is responsible for network-centric operation and is aware of many features such as topology, technology, and operator policy. As an optional component, it can use the service model as input and is not required at all if the control component delegates its control operations to the configuration component.

In Section 7 we provide some example of translation of service provisioning requests to router configuration lines as an illustration. In the YANG based ecosystem, it is expected that NETCONF and YANG will be used between the configuration component and network elements to configure the requested service on those elements.

In this framework, it is expected that YANG models will be used for configuring service components on network elements. There will be a strong relationship between the abstracted view provided by this service model and the detailed configuration view that will be provided by specific configuration models for network elements such as those defined in [I-D.ietf-bess-l2vpn-yang] and [I-D.ietf-bess-evpn-yang]. Service components needing configuration on network elements in support of the service model defined in this document include:

- Network Instance definition including VPN policy expression.
- Physical interface.
- Ethernet layer (VLAN ID).
- QoS: classification, profiles, etc.
- Ethernet Service OAM Support.

7. Service Model Usage Example

As explained in Section 4, this service model is intended to be instantiated at a management layer and is not intended to be used directly on network elements. The management system serves as a central point of configuration of the overall service.

This section provides an example on how a management system can use this model to configure an L2VPN service on network elements.

The example is to provide a VPN service for 3 sites using point-to-point VPWS and a Hub and Spoke VPN service topology as shown in Figure 5. Loadbalancing is not considered in this case.

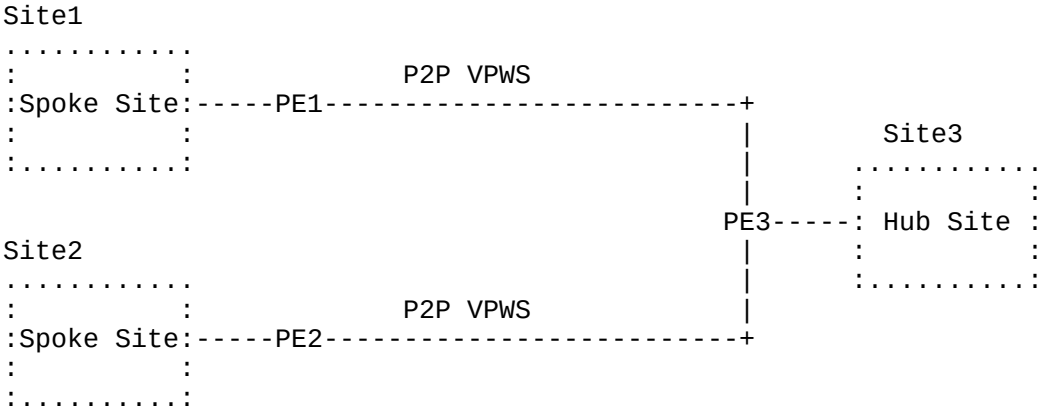


Figure 5: Reference Network for Simple Example

The following XML describes the overall simplified service configuration of this VPN.

```
<?xml version="1.0"?>
<l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
  <vpn-services>
      <vpn-service>
       <vpn-id>12456487</vpn-id>
        <vpn-svc-type>vpws</vpn-svc-type>
        <svc-topo>hub-spoke</svc-topo>
        <ce-vlan-preservation>true</ce-vlan-preservation>
        <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
       </vpn-service>
      <vpn-service>
        <vpn-id>12456488</vpn-id>
        <vpn-svc-type>vpws</vpn-svc-type>
        <svc-topo>hub-spoke</svc-topo>
        <ce-vlan-preservation>true</ce-vlan-preservation>
        <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
       </vpn-service>
  </vpn-services>
</l2vpn-svc>
```

When receiving the request for provisioning the VPN service, the management system will internally (or through communication with another OSS component) allocates VPN route-targets. In this specific case two Route Targets (RTs) will be allocated (100:1 for Hubs and 100:2 for Spokes). The output below describes the configuration of Spoke Site1.

```
<?xml version="1.0"?>
<l2vpn-svc xmlns="urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc">
 <vpn-services>
  <vpn-service>
   <vpn-id>12456487</vpn-id>
   <svc-topo>hub-spoke</svc-topo>
   <ce-vlan-preservation>true</ce-vlan-preservation>
   <ce-vlan-cos-preservation>true</ce-vlan-cos-preservation>
  </vpn-service>
 </vpn-services>
 <sites>
   <site>
      <site-id>Spoke_Site1</site-id>
         <locations>
           <location>
            <location-id>NY1</location-id>
            <city>NY</city>
            <country-code>US</country-code>
           </location>
          </locations>
          <site-network-accesses>
             <site-network-access>
                <network-access-id>Spoke_UNI-Site1</network-access-id>
                  <access-diversity>
                    <groups>
                      <group>
                        <group-id>20</group-id>
                      </group>
                    </groups>
                  </access-diversity>
                    <connection>
                     <encapsulation-type>vlan</encapsulation-type>
                      <tagged-interface>
                      <dot1q-vlan-tagged>
                        <cvlan-id>17</cvlan-id>
                      </dot1q-vlan-tagged>
                      </tagged-interface>
                      <l2cp-control>
                        <stp-rstp-mstp>tunnel</stp-rstp-mstp>
                        <lldp>true</lldp>
                      </l2cp-control>
```

```
                    </connection>
                    <service>
                      <svc-bandwidth>
                        <bandwidth>
                        <direction>input-bw</direction>
                        <type>bw-per-cos</type>
                        <cir>450000000</cir>
                        <cbs>20000000</cbs>
                        <eir>1000000000</eir>
                        <ebs>200000000</ebs>
                        </bandwidth>
                      </svc-bandwidth>
                      <carrierscarrier>
                       <signaling-type>bgp</signaling-type>
                      </carrierscarrier>
                    </service>
                     <vpn-attachment>
                       <vpn-id>12456487</vpn-id>
                       <site-role>spoke-role</site-role>
                     </vpn-attachment>
                   </site-network-access>
                 </site-network-accesses>
                 <management>
                   <type>provider-managed</type>
                 </management>
               </site>
        </sites>
    </l2vpn-svc>
```

When receiving the request for provisioning the Spoke1 site, the management system MUST allocate network resources for this site. It MUST first determine the target network elements to provision the access, and especially the PE router (and may be an aggregation switch). As described in Section 5.3.1, the management system SHOULD use the location information and MUST use the access-diversity constraint to find the appropriate PE. In this case, we consider Spoke1 requires PE diversity with Hub and that management system allocate PEs based on lowest distance. Based on the location information, the management system finds the available PEs in the nearest area of the customer and picks one that fits the access-diversity constraint.

When the PE is chosen, the management system needs to allocate interface resources on the node. One interface is selected from the PE available pool. The management system can start provisioning the PE node by using any mean (NETCONF, CLI, ...). The management system will check if a VSI is already present that fits the needs. If not, it will provision the VSI: the Route Distinguisher will come from the

internal allocation policy model, and the route-targets come from the vpn-policy configuration of the site (management system allocated some RTs for the VPN). As the site is a Spoke site (site-role), the management system knows which RT must be imported and exported. As the site is provider managed, some management route-targets may also be added (100:5000). Standard provider VPN policies MAY also be added in the configuration.

Example of generated PE configuration:

```
l2vpn vsi context one
  vpn id 12456487
     autodiscovery bgp signaling bgp
     ve id 1001      <----identify the PE routers within the VPLS domain
     ve range 50    <---- VE size
     route-distinguisher 100:3123234324
     route-target import 100:1
     route-target import 100:5000    <---- Standard SP configuration
     route-target export 100:2                for provider managed CE
   !
```

When the VSI has been provisioned, the management system can start configuring the access on the PE using the allocated interface information. The tag or VLAN (e.g., service instance tag)is chosen by the management system. One tag will be picked from an allocated subnet for the PE, another will be used for the CE configuration. LACP protocols will also be configured between PE and CE and due to provider managed model, the choice is up to service provider. This choice is independent of the LACP protocol chosen by customer.

Example of generated PE configuration:

```
!
bridge-domain 1
 member Ethernet0/0 service-instance 100
 member vsi one
!
l2 router-id 198.51.100.1
!
l2 router-id 2001:db8::10:1/64
!

interface Ethernet0/0
 no ip address
 service instance 100 ethernet
encapsulation dot1q 100
 !

!
router bgp 1
 bgp log-neighbor-changes
 neighbor 198.51.100.4 remote-as 1
 neighbor 198.51.100.4 update-source Loopback0
 !
 address-family l2vpn vpls
  neighbor 198.51.100.4 activate
  neighbor 198.51.100.4 send-community extended
  neighbor 198.51.100.4 suppress-signaling-protocol ldp
  neighbor 2001:db8::0a10:4 activate
  neighbor 2001:db8::0a10:4 send-community extended
 exit-address-family

!
interface vlan 100 <-- Associating the Attachment
  no ip address       Circuit with the MAC-VRF at the PE
  xconnect vsi PE1-VPLS-A
!
vlan 100
  state active
```

As the CE router is not reachable at this stage, the management system can produce a complete CE configuration that can be uploaded to the node by manual operation before sending the CE to customer premise. The CE configuration will be built as for the PE. Based on the CE type (vendor/model) allocated to the customer and bearer information, the management system knows which interface must be configured on the CE. PE-CE link configuration is expected to be handled automatically using the service provider OSS as both resources are managed internally. CE to LAN interface parameters like dot1Q tag are derived from the ethernet-connection taking into

account how management system distributes dot1Q tag between PE and CE within subnet. This will allow to produce a plug'n'play configuration for the CE.

Example of generated CE configuration:

```
interface Ethernet0/1
  switchport trunk allowed vlan none
  switchport mode trunk
  service instance 100 ethernet
  encapsulation default
  l2protocol forward cdp
  xconnect 203.0.113.1 100 encapsulation mpls
!
```

8. YANG Module

```
<CODE BEGINS> file "ietf-l2vpn-svc@2018-04-03.yang"
module ietf-l2vpn-svc {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc";
  prefix l2vpn-svc;

  import ietf-inet-types {
    prefix inet;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-netconf-acm {
    prefix nacm;
  }

  organization
    "IETF L2SM Working Group.";
  contact
    "WG List: l2sm@ietf.org
     Editor: giuseppe.fioccola@telecomitalia.it";
  description
    "The YANG module defines a generic service configuration
     model for Layer 2 VPN services common across all of the
     vendor implementations.";

  revision 2018-04-03 {
    description
      "Initial revision";
    reference
      "RFC xxxx: A YANG Data Model for L2VPN Service
```

```
       Delivery.";
  }

  feature carrierscarrier {
    description
      "Enables support of CsC.";
  }

  feature ethernet-oam {
    description
      "Enables support of ethernet OAM.";
  }

  feature extranet-vpn {
    description
      "Enable the Support of Extranet VPN.";
  }

  feature l2cp-control {
    description
      "Enable the Support of L2CP control.";
  }

  feature input-bw {
    description
      "Enable the suppport of Input Bandwidth in a VPN.";
  }

  feature output-bw {
    description
      "Enable the support of Output Bandwidth in a VPN";
  }

  feature uni-list {
    description
      "Enable the support of UNI list in a VPN.";
  }

  feature cloud-access {
    description
      "Allow VPN to connect to a Cloud Service
       provider or an internet service provider.";
  }

  feature oam-3ah {
    description
      "Enables the support of OAM 802.3ah.";
  }
```

```
  feature micro-bfd {
    description
      "Enables the support of Micro-BFD.";

  }

  feature bfd {
    description
      "Enables the support of BFD.";
  }

  feature signaling-options {
    description
      "Enable the support of signaling option.";
  }

  feature site-diversity {
    description
      "Enables the support of site diversity constraints in a VPN.";
  }

  feature encryption {
    description
      "Enables support of encryption.";
  }

  feature always-on {
    description
      "Enables support for always-on access
       constraint.";
  }

  feature requested-type {
    description
      "Enables support for requested-type access
       constraint.";
  }

  feature bearer-reference {
    description
      "Enables support for bearer-reference access
       constraint.";
  }

  feature qos {
    description
      "Enables support of Class of Services.";
  }
```

```
  feature qos-custom {
    description
      "Enables support of custom qos profile.";

  }

  feature lag-interface {
    description
      "Enable lag-interface.";
  }

  feature vlan {
    description
      "Enable the support of VLAN.";
  }

  feature dot1q {
    description
      "Enable the support of Dot1Q.";
  }

  feature sub-inf {
    description
      "Enable the support of Sub Interface.";
  }

  feature qinq {
    description
      "Enable the support of QinQ.";
  }

  feature qinany {
    description
      "Enable the support of QinAny.";
  }

  feature vxlan {
    description
      "Enable the support of VXLAN.";
  }

  feature lan-tag {
    description
      "Enables LAN Tag support in a VPN.";
  }

  feature target-sites {
    description
```

```
      "Enables support of the 'target-sites' match flow parameter.";
  }

  feature bum {

    description
      "Enables broadcast,unknown-unicast,multicast
       capabilities in a VPN.";
  }

  feature mac-loop-prevention {
    description
      "Enables MAC Loop prevention capability in a VPN.";
  }

  feature lacp {
    description
      "Enables LACP capability in a VPN.";
  }

  feature mac-addr-limit{
    description
      "Enables MAC Address Limit capability in a VPN.";
  }

  feature acl {
    description
      "Enables ACL capability in a VPN. ";
  }

  feature cfm {
    description
      "Enables cfm 802.1 ag capability in a VPN.";
  }

  feature y-1731 {
    description
      "Enable Y.1731 capability in a VPN.";
 }

  typedef svc-id {
    type string;
    description
      "Defines a type of service component identifier.";
  }

  typedef ccm-priority-type {
    type uint8 {
```

```
      range "0..7";
    }
    description
      "A 3 bits priority value to be used in the VLAN tag,
       if present in the transmitted frame.";

  }

  typedef control-mode {
    type enumeration {
      enum "peer" {
        description
          "Peer mode, i.e., participate in the protocol towards the CE.
           Peering is common for LACP  and E-LMI and occasionally
           for LLDP. For virtual private services the Subscriber
           can also request that the Service Provider peer
           spanning tree.";
      }
      enum "tunnel" {
        description
          "Tunnel mode,i.e.,pass to the egress or destination site.
           For EPL, the expectation is that L2CP frames are tunneled.";
      }
      enum "discard" {
        description
          "Discard mode,i.e.,discard the frame.";
      }
    }
    description
      "Defining a type of the control mode on L2CP protocols.";
  }

  typedef neg-mode {
    type enumeration {
      enum "full-duplex" {
        description
          "Defining Full duplex mode";
      }
      enum "auto-neg" {
        description
          "Defining Auto negotiation mode";
      }
    }
    description
      "Defining a type of the negotiation mode";
  }

  identity site-network-access-type {
```

```
    description
      "Base identity for site-network-access type.";
  }

  identity point-to-point {
    base site-network-access-type;

    description
      "Identity for point-to-point connection.";
  }

  identity multipoint {
    base site-network-access-type;
    description
      "Identity for multipoint connection.
       Example: Ethernet broadcast segment.";
  }

  identity tag-type {
    description
      "Base identity from which all tag types
       are derived from";
  }

  identity c-vlan {
    base tag-type;
    description
      "A Customer-VLAN tag, normally using the 0x8100
       Ethertype";
  }

  identity s-vlan {
    base tag-type;
    description
      "A Service-VLAN tag.";
  }

  identity c-s-vlan {
    base tag-type;
    description
      "Using both Customer-VLAN tag and Service-VLAN tag.";
  }

  identity multicast-tree-type {
    description
      "Base identity for multicast tree type.";
  }
```

```
  identity ssm-tree-type {
    base multicast-tree-type;
    description
      "Identity for SSM tree type.";
  }

  identity asm-tree-type {

    base multicast-tree-type;
    description
      "Identity for ASM tree type.";
  }

  identity bidir-tree-type {
    base multicast-tree-type;
    description
      "Identity for bidirectional tree type.";
  }

  identity mapping-type {
    description
      "Identity mapping-type";
  }

  identity static-mapping {
    base mapping-type;
    description
      "Identity for static mapping, i.e.,attach the interface
       to the Multicast group as static member";
  }

  identity dynamic-mapping {
    base mapping-type;
    description
      "Identity for dynamic mapping, i.e.,interface was added
       to the Multicast group as a result of snooping";
  }

  identity tf-type {
    description
      "Identity traffic-type";
  }

  identity multicast-traffic {
    base tf-type;
    description
      "Identity for multicast traffic";
  }
```

```
  identity broadcast-traffic {
    base tf-type;
    description
      "Identity for broadcast traffic";
  }

  identity unknown-unicast-traffic {

    base tf-type;
    description
      "Identity for unknown unicast traffic";
  }

  identity encapsulation-type {
    description
      "Identity for encapsulation type";
  }

  identity ethernet {
    base encapsulation-type;
    description
      "Identity for ethernet type";
  }

  identity vlan {
    base encapsulation-type;
    description
      "Identity for VLAN  type";
  }

  identity carrierscarrier-type {
    description
      "Identity of carrierscarrier";
  }

  identity ldp {
    base carrierscarrier-type;
    description
      "Use LDP as the signalling protocol
       between the PE and the CE.";
  }

  identity bgp {
    base carrierscarrier-type;
    description
      "Use BGP (as per RFC 3107) as the signalling protocol
       between the PE and the CE.
       In this case, BGP must also be configured as
```

```
       the routing protocol.";
  }

  identity eth-inf-type {
    description
      "Identity of Ethernet Interface Type.";
  }

  identity tagged {
    base eth-inf-type;
    description
      "Identity of tagged Interface type.";
  }

  identity untagged {
    base eth-inf-type;
    description
      "Identity of untagged Interface type.";
  }

  identity lag {
    base eth-inf-type;
    description
      "Identity of LAG Interface type";
  }

  identity bw-type {
    description
      "Identity of bandwidth";
  }

  identity bw-per-cos {
    base bw-type;
    description
      "Bandwidth is per cos";
  }

  identity bw-per-port {
    base bw-type;
    description
      "Bandwidth is per site network access";
  }

  identity bw-per-site {
    base bw-type;
    description
      "Bandwidth is per site. It is applicable to
       all the site network accesses within the site.";
```

```
  }

  identity bw-per-svc {
    base bw-type;
    description
      "Bandwidth is per VPN service";
  }

  identity site-vpn-flavor {
    description
      "Base identity for the site VPN service flavor.";
  }

  identity site-vpn-flavor-single {
    base site-vpn-flavor;
    description
      "Identity for the site VPN service flavor.
       Used when the site belongs to only one VPN.";
  }

  identity site-vpn-flavor-multi {
    base site-vpn-flavor;
    description
      "Identity for the site VPN service flavor.
       Used when a logical connection of a site
       belongs to multiple VPNs.";
  }

  identity site-vpn-flavor-nni {
    base site-vpn-flavor;
    description
      "Identity for the site VPN service flavor.
       Used to describe an NNI option A connection.";
  }

  identity service-type {
    description
      "Base Identity of service type.";
  }

  identity vpws {
    base service-type;
    description
      "point-to-point Virtual Private Wire Services(VPWS) type.";
  }

  identity pwe3 {
    base service-type;
```

```
    description
      "Pseudo-Wire Emulation Edge to
       Edge (PWE3) Service type.";
  }

  identity ldp-l2tp-vpls {
    base service-type;
    description

      "LDP based or L2TP based multipoint Virtual Private LAN
       services (VPLS) Service Type.This VPLS uses LDP-signaled
       Pseudowires or L2TP signaled Pseudowires.";
  }

  identity bgp-vpls {
    base service-type;
    description
      "BGP based multipoint Virtual Private LAN services (VPLS)
       Service Type. This VPLS uses a Border Gateway Protocol
       (BGP) control plane as described in RFC4761 and RFC6624.";
  }

  identity vpws-evpn {
    base service-type;
    description
      "VPWS Service Type using Ethernet VPN(EVPN)
      specified in RFC 7432.";
  }

  identity pbb-evpn {
    base service-type;
    description
      "PBB Service Type using Ethernet VPN(EVPN)
      specified in RFC 7432.";
  }

  identity bundling-type {
    description
      "This is base identity for Bundling type. It supports
       multiple CE-VLAN associated with L2VPN service or
       all CE-VLANs associated with L2VPN service.";
  }

  identity multi-svc-bundling {
    base bundling-type;
    description
      "Identity for multiple service bundling,i.e.,
       multiple CE-VLAN IDs can be associated with an
```

```
       L2VPN Service at site.";
  }

  identity one2one-bundling {
    base bundling-type;
    description
      "Identity for one to one service bundling,i.e.,
       Each L2VPN can be associated with only one CE-VLAN IDs
       at site.";

  }

  identity all2one-bundling {
    base bundling-type;
    description
      "Identity for all to one bundling,i.e.,all CE-VLAN IDs
       are mapped to one L2VPN Service";
  }

  identity color-id {
    description
      "base identity of color id";
  }

  identity color-id-cvlan {
    base color-id;
    description
      "Identity of color id base on  CVLAN ";
  }

  identity cos-id {
    description
      "Identity of class of service id";
  }

  identity cos-id-pcp {
    base cos-id;
    description
      "Identity of cos id based on  PCP";
  }

  identity cos-id-dscp {
    base cos-id;
    description
      "Identity of cos id based on  DSCP";
  }

  identity color-type {
```

```
    description
      "Identity of color types";
  }

  identity green {
    base color-type;
    description
      "Identity of green type";
  }

  identity yellow {
    base color-type;
    description
      "Identity of yellow type";
  }

  identity red {
    base color-type;
    description
      "Identity of red type";
  }

  identity policing {
    description
      "Identity of policing type";
  }

  identity one-rate-two-color {
    base policing;
    description
      "Identity of one-rate, two-color (1R2C).";
  }

  identity two-rate-three-color {
    base policing;
    description
      "Identity of two-rate, three-color (2R3C).";
  }

  identity bum-type {
    description
      "Identity of BUM type.";
  }

  identity broadcast {
    base bum-type;
    description
      "Identity of broadcast.";
```

```
  }

  identity unicast {
    base bum-type;
    description
      "Identity of unicast";
  }

  identity multicast {
    base bum-type;

    description
      "Identity of multicast.";
  }

  identity loop-prevention-type {
    description
      "Identity of loop prevention.";
  }

  identity shut {
    base loop-prevention-type;
    description
      "Identity of shut protection.";
  }

  identity trap {
    base loop-prevention-type;
    description
      "Identity of trap protection.";
  }

  identity lacp-state {
    description
      "Identity of LACP state.";
  }

  identity lacp-on {
    base lacp-state;
    description
      "Identity of LCAP on.";
  }

  identity lacp-off {
    base lacp-state;
    description
      "Identity of LACP off";
  }
```

```
  identity lacp-mode {
    description
      "Identity of LACP mode";
  }

  identity lacp-passive {
    base lacp-mode;
    description
      "Identity of LACP passive";
  }

  identity lacp-active {
    base lacp-mode;
    description
      "Identity of LACP active";
  }

  identity lacp-speed {
    description
      "Identity of LACP speed";
  }

  identity lacp-fast {
    base lacp-speed;
    description
      "Identity of LACP fast";
  }

  identity lacp-slow {
    base lacp-speed;
    description
      "Identity of LACP slow";
  }

  identity bw-direction {
    description
      "Identity for bandwidth direction";
  }

  identity input-bw {
    base bw-direction;
    description
      "Identity for input bandwidth";
  }

  identity output-bw {
    base bw-direction;
    description
```

```
      "Identity for output bandwidth";
  }

  identity management {
    description
      "Base identity for site management scheme.";
  }

  identity co-managed {
    base management;
    description

      "Identity for co-managed site.";
  }

  identity customer-managed {
    base management;
    description
      "Identity for customer managed site.";
  }

  identity provider-managed {
    base management;
    description
      "Identity for provider managed site.";
  }

  identity address-family {
    description
      "Identity for an address family.";
  }

  identity ipv4 {
    base address-family;
    description
      "Identity for IPv4 address family.";
  }

  identity ipv6 {
    base address-family;
    description
      "Identity for IPv6 address family.";
  }

  identity vpn-topology {
    description
      "Base identity for VPN topology.";
  }
```

```
  identity any-to-any {
    base vpn-topology;
    description
      "Identity for any to any VPN topology.";
  }

  identity hub-spoke {
    base vpn-topology;
    description
      "Identity for Hub’n’Spoke VPN topology.";
  }

  identity hub-spoke-disjoint {
    base vpn-topology;
    description
      "Identity for Hub’n’Spoke VPN topology
       where Hubs cannot talk between each other.";
  }

  identity site-role {
    description
      "Base identity for site type.";
  }

  identity any-to-any-role {
    base site-role;
    description
      "Site in an any to any L2VPN.";
  }

  identity spoke-role {
    base site-role;
    description
      "Spoke Site in a Hub-and-Spoke L2VPN.";
  }

  identity hub-role {
    base site-role;
    description
      "Hub Site in a Hub-and-Spoke L2VPN.";
  }

  identity pm-type {
    description
      "Performance monitor type";
  }

  identity loss {
```

```
    base pm-type;
    description
      "Loss measurement";
  }

  identity delay {
    base pm-type;
    description
      "Delay measurement";
  }

  identity fault-alarm-defect-type {

    description
      "Indicating the alarm priority defect";
  }

  identity remote-rdi {
    base fault-alarm-defect-type;
    description
      "Indicates the aggregate health
       of the remote MEPs.";
  }

  identity remote-mac-error {
    base fault-alarm-defect-type;
    description
      "Indicates that one or more of the remote MEPs is
       reporting a failure in its Port Status TLV or
       Interface Status TLV.";
  }

  identity remote-invalid-ccm {
    base fault-alarm-defect-type;
    description
      "Indicates that at least one of the Remote MEP
       state machines is not receiving valid CCMs
       from its remote MEP.";
  }

  identity invalid-ccm {
    base fault-alarm-defect-type;
    description
      "Indicates that one or more invalid CCMs has been
       received and that 3.5 times that CCMs transmission
       interval has not yet expired.";
  }
```

```
  identity cross-connect-ccm {
    base fault-alarm-defect-type;
    description
      "Indicates that one or more cross connect CCMs has been
       received and that 3.5 times of at least one of those
       CCMs transmission interval has not yet expired.";
  }

  identity frame-delivery-mode {
    description
      "Delivery types";
  }

  identity discard {
    base frame-delivery-mode;
    description
      "Service Frames are discarded.";
  }

  identity unconditional {
    base frame-delivery-mode;
    description
      "Service Frames are unconditionally
       delivered to the destination.";
  }

  identity unknown-discard {
    base frame-delivery-mode;
    description
      "Service Frame are conditionally
       delivered to the destination site and
       the packet with unknown destination address
       will be discarded.";
  }

  identity placement-diversity {
    description
      "Base identity for site placement
       constraints.";
  }

  identity bearer-diverse {
    base placement-diversity;
    description
      "Identity for bearer diversity.
       The bearers should not use common elements.";
  }
```

```
  identity pe-diverse {
    base placement-diversity;
    description
      "Identity for PE diversity";
  }

  identity pop-diverse {
    base placement-diversity;
    description
      "Identity for POP diversity";
  }

  identity linecard-diverse {
    base placement-diversity;
    description
      "Identity for linecard diversity";
  }

  identity same-pe {
    base placement-diversity;
    description
      "Identity for having sites connected
       on the same PE";
  }

  identity same-bearer {
    base placement-diversity;
    description
      "Identity for having sites connected
       using the same bearer";
  }

  identity tagged-inf-type {
    description
      "Identity for the tagged
       interface type.";
  }

  identity priority-tagged {
    base tagged-inf-type;
    description
      "This identity the priority-tagged interface.";
  }

  identity qinq {
    base tagged-inf-type;
    description
      "Identity for the qinq tagged interface.";
```

```
  }

  identity dot1q {
    base tagged-inf-type;
    description
      "Identity for dot1q vlan tagged interface.";
  }

  identity qinany {
    base tagged-inf-type;
    description
      "Identity for qinany tagged inteface.";
  }

  identity vxlan {
    base tagged-inf-type;
    description
      "Identity for vxlan tagged inteface.";
  }

  identity provision-model {
    description
      "base identity for provision model.";
  }

  identity single-side-provision {
    description
      "Identity for single side provisioning with discovery.";
  }

  identity doubled-side-provision {
    description
      "Identity for double side provisioning.";
  }

  identity mac-learning-mode {
    description
      "MAC learning mode";
  }

  identity data-plane {
    base mac-learning-mode;
    description
      "User MAC addresses are learned through ARP broadcast.";
  }

  identity control-plane {
    base mac-learning-mode;
```

```
    description
      "User MAC addresses are advertised through EVPN-BGP";
  }

  identity vpn-policy-filter-type {
    description
      "Base identity for filter type.";
  }

  identity lan {
    base vpn-policy-filter-type;
    description
      "Identity for lan tag filter type.";
  }

  identity mac-action {
    description
      "Base identity for MAC action.";
  }

  identity drop {
    base mac-action;
    description
      "Identity for packet drop.";
  }

  identity flood {
    base mac-action;
    description
      "Identity for packet flooding.";
  }

  identity warning {
    base mac-action;
    description
      "Identity for sending a warning log message.";
  }

  identity qos-profile-direction {
    description
      "Base identity for qos profile direction.";
  }

  identity site-to-wan {
    base qos-profile-direction;
    description
      "Identity for Site to WAN direction.";
  }
```

```
  identity wan-to-site {
    base qos-profile-direction;
    description
      "Identity for WAN to Site direction.";
  }

  identity bidirectional {
    base qos-profile-direction;
    description
      "Identity for both WAN to Site direction
       and Site to WAN direction.";
  }

  identity vxlan-peer-mode {
    description
      "Base identity for vxlan peer mode.";
  }

  identity static-mode {
    base vxlan-peer-mode;
    description
      "Identity for the vxlan access in static mode.";
  }

  identity bgp-mode {
    base vxlan-peer-mode;
    description
      "Identity for the vxlan access by bgp evpn learning.";
  }

  identity customer-application {
    description
      "Base identity for customer application.";
  }

  identity web {
    base customer-application;
    description
      "Identity for Web application (e.g., HTTP, HTTPS).";
  }

  identity mail {
    base customer-application;
    description
      "Identity for mail application.";
  }

  identity file-transfer {
```

```
    base customer-application;
    description
      "Identity for file transfer application
      (e.g., FTP, SFTP).";
  }

  identity database {
    base customer-application;
    description
      "Identity for database application.";
  }

  identity social {
    base customer-application;
    description
      "Identity for social-network application.";
  }

  identity games {
    base customer-application;
    description
      "Identity for gaming application.";
  }

  identity p2p {
    base customer-application;
    description
      "Identity for peer-to-peer application.";
  }

  identity network-management {
    base customer-application;
    description
      "Identity for management application
       (e.g., Telnet, syslog, SNMP).";

  }

  identity voice {
    base customer-application;
    description
      "Identity for voice application.";
  }

  identity video {
    base customer-application;
    description
      "Identity for video conference application.";
```

```
  }

  identity embb {
    base customer-application;
    description
      "Identity for enhanced Mobile Broadband(eMBB)
       application. Note that eMBB application demands
       the network performance with wide variety of
       characteristics such as data rate, latency,
       loss rate, reliability and many other parameters.";
  }

  identity urllc {
    base customer-application;
    description
      "Identity for Ultra-Reliable and Low Latency
       Communications (URLLC) application. Note that
       URLLC application demands the network performance
       with wide variety of characteristics such as latency,
       reliability and many other parameters.";
  }

  identity mmtc {
    base customer-application;
    description
      "Identity for massive Machine Type
       Communications (mMTC) application. Note that
       mMTC application demands the network performance
       with wide variety of characteristics such as data
       rate, latency, loss rate, reliability and many
       other parameters.";
  }

  grouping site-acl {
    container access-control-list {
      if-feature acl;
      list mac {
        key "mac-address";
        leaf mac-address {
          type yang:mac-address;
          description
            "MAC address.";
        }
        description
          "List for MAC.";
      }
      description
        "Container for access control List.";
```

```
    }
    description
      "This grouping defines Access Control List.";
  }

  grouping site-bum {
    container broadcast-unknown-unicast-multicast {
      if-feature bum;
      leaf multicast-site-type {
        type enumeration {
          enum "receiver-only" {
            description
              "The site only has receivers.";
          }
          enum "source-only" {
            description
              "The site only has sources.";
          }
          enum "source-receiver" {
            description
              "The site has both sources and receivers.";
          }
        }

        default "source-receiver";
        description
          "Type of multicast site.";
      }
      list multicast-gp-address-mapping {
        key "id";
        leaf id {
          type uint16;
          description
            "Unique identifier for the mapping.";
        }
        leaf vlan-id {
          type uint16 {
            range "0..1024";
          }
          mandatory true;
          description
            "The VLAN ID of the Multicast group.
             The range of 12 bit VLAN ID is 0 to 1024.";
        }
        leaf mac-gp-address {
          type yang:mac-address;
          mandatory true;
          description
```

```
            "the MAC address of the Multicast group.";
        }
        leaf port-lag-number {
          type uint32;
          description
            "the ports/LAGs belonging to the Multicast group.";
        }
        description
          "List of Port to group mappings.";
      }
      leaf bum-overall-rate {
        type uint64;
        units "bps";
        description
          "overall rate for BUM.";
      }
      list bum-rate-per-type {
        key "type";
        leaf type {
          type identityref {
            base bum-type;
          }
          description
            "BUM type.";
        }
        leaf rate {
          type uint64;
          units "bps";
          description
            "rate for BUM.";
        }
        description
          "List of rate per type.";
      }
      description
        "Container of broadcast, unknown unicast, and multicast
         configurations.";
    }
    description
      "Grouping for broadcast, unknown unicast, and multicast.";
  }

  grouping site-mac-loop-prevention {
    container mac-loop-prevention {
      if-feature mac-loop-prevention;
      leaf protection-type {
        type identityref {
          base loop-prevention-type;
```

```
        }
        default "trap";
        description
          "Protection type. By default, the protection
           type is trap protection type.";
      }
      leaf frequency {
        type uint32;
        default "5";
        description
          "The number of times to detect MAC duplication.
           When duplicate-MAC situation has occurred and the
           duplicated MAC is added into to a duplicate-MAC
           list. By default, the number of times is 5.";
      }
      leaf retry-timer {
        type uint32;
        units "seconds";
        description
          "The retry timer. When retry timer expires,
           the duplicated MAC will be flushed from
           the MAC-VRF. ";
      }
      description
        "Container of MAC loop prevention.";
    }
    description
      "Grouping for MAC loop prevention.";
  }

  grouping site-service-qos-profile {
    container qos {
      if-feature qos;
      container classification-policy {
        list rule {
          key "id";
          ordered-by user;
          leaf id {
            type string;
            description
              "A description identifying qos classification
               policy rule.";
          }
          choice match-type {
            default "match-flow";
            case match-flow {
    container match-flow {
      leaf dscp {
```

```
        type inet:dscp;
        description
          "DSCP value.";
      }
      leaf dot1q {
        type uint16;

        description
          "802.1q matching. It is VLAN Tag added into frame.";
      }
      leaf pcp {
        type uint8 {
          range "0 .. 7";
        }
        description
          "PCP value.";
      }
      leaf src-mac {
        type yang:mac-address;
        description
          "Source MAC";
      }
      leaf dst-mac {
        type yang:mac-address;
        description
          "Destination MAC.";
      }
      leaf color-type {
        type identityref {
          base color-type;
        }
        description
          "Color Types.";
      }
      leaf-list target-sites {
        if-feature target-sites;
        type svc-id;
        description
          "Identify a site as traffic destination.";
      }
      leaf any {
        type empty;
        description
          "Allow all.";
      }
      leaf vpn-id {
        type svc-id;
        description
```

```
          "Reference to the target VPN.";
      }
      description
        "Describe flow matching criteria.";
    }
          }
          case match-application {
            leaf match-application {
              type identityref {
                base customer-application;
              }
              description
                "Defines the application to match.";
            }
          }
          description
            "Choice for classification";
        }
        leaf target-class-id {
          type string;
          description
            "Identification of the class of service.
             This identifier is internal to the
             administration.";
        }
        description
          "List of marking rules.";
      }
      description
        "Configuration of the traffic classification policy.";
    }
    container qos-profile {
      choice qos-profile {
        description
          "Choice for QoS profile.
           Can be standard profile or customized profile.";
        case standard {
          description
            "Standard QoS profile.";
          leaf profile {
            type leafref {
              path "/l2vpn-svc/vpn-profiles/"
               +"valid-provider-identifiers"
                +"/qos-profile-identifier";
            }
            description
              "QoS Profile to be used.";
          }
```

```
          }
          case custom {
            description
              "Customized QoS profile.";
            container classes {
              if-feature qos-custom;
              list class {
                key "class-id";
                leaf class-id {
                  type string;
                  description
                    "Identification of the class of
                     service. This identifier is internal
                     to the administration.";
                }
                leaf direction {
                  type identityref {
                    base qos-profile-direction;
                  }
                  default "bidirectional";
                  description
                    "The direction which QoS profile is applied to.
                     By default,the direction is bidirectional.";
                }
                leaf policing {
                  type identityref {
                    base policing;
                  }
                  default "one-rate-two-color";
                  description
                    "The policing can be either one-rate,
                     two-color (1R2C) or two-rate, three-color
                     (2R3C). By default, the policing is on rate
                     two color.";

                }
                leaf byte-offset {
                  type uint16;
                  description
                    "For not including extra VLAN tags in the QoS
                     calculation.";
                }
                container frame-delay {
                  choice flavor {
                    case lowest {
                      leaf use-lowest-latency {
                        type empty;
                        description
```

```
                    "The traffic class should use
                     the lowest delay path.";
                }
              }
              case boundary {
                leaf delay-bound {
                  type uint16;
                  units "msec";
                  description
                    "The traffic class should use
                     a path with a defined maximum
                     delay.";
                }
              }
              description
                "Delay constraint on the traffic
                 class.";
            }
            description
              "Delay constraint on the traffic
               class.";
          }
          container frame-jitter {
            choice flavor {
              case lowest {
                leaf use-lowest-jitter {
                  type empty;
                  description
                    "The traffic class should use
                     the lowest jitter path.";
                }
              }
              case boundary {
                leaf delay-bound {
                  type uint32;

                  units "usec";
                  description
                    "The traffic class should use
                     a path with a defined maximum
                     jitter.";
                }
              }
              description
                "Jitter constraint on the traffic
                 class.";
            }
            description
```

```
                    "Jitter constraint on the traffic
                     class.";
                }
                container frame-loss {
                  leaf rate {
                    type decimal64 {
                      fraction-digits 2;
                      range "0..100";
                    }
                    units "percent";
                    description
                      "Frame Loss rate constraint on the traffic
                      class.";
                  }
                  description
                    "Container for frame loss rate.";
                }
                container bandwidth {
                  leaf guaranteed-bw-percent {
                    type decimal64 {
                      fraction-digits 5;
                      range "0..100";
                    }
                    units "percent";
                    mandatory true;
                    description
                      "To be used to define the guaranteed bandwidth
                       as a percentage of the available service
                       bandwidth.";
                  }
                  leaf end-to-end {
                    type empty;
                    description
                      "Used if the bandwidth reservation
                       must be done on the MPLS network too.";
                  }
                  description
                    "Bandwidth constraint on the traffic class.";
                }
                description
                  "List of class of services.";
              }
              description
                "Container for list of class of services.";
            }
          }
        }
        description
```

```
          "Qos profile configuration.";
      }
      description
        "QoS configuration.";
    }
    description
      "This grouping defines QoS parameters
       for a site";
  }

  grouping site-service-mpls {
    container carrierscarrier {
      if-feature carrierscarrier;
      leaf signaling-type {
        type identityref {
          base carrierscarrier-type;
        }
        default "bgp";
        description
          "Carrierscarrier. By default,the signaling type is bgp.";
      }
      description
        "Container for carrierscarrier";
    }
    description
      "Grouping for carrierscarrier";
  }

  container l2vpn-svc {
    container vpn-profiles {
    container valid-provider-identifiers {
      leaf-list cloud-identifier {
        if-feature cloud-access;
          type string;
          description
            "Identification of public cloud service
             or internet service. Local administration
             meaning.";
      }
      leaf-list qos-profile-identifier {
          type string;
          description
            "Identification of the QoS Profile to be used.
             Local administration meaning.";
      }
      leaf-list bfd-profile-identifier {
          type string;
          description
```

```
            "Identification of the SP BFD Profile to be used.
             Local administration meaning.";
      }
      leaf-list remote-carrier-identifier {
          type string;
          description
            "Identification of the remote carrier name to be used.
             It can be L2VPN partner, Data center service provider
             or private cloud service provider. Local administration
             meaning.";
      }
      nacm:default-deny-write;
      description
        "Container for Valid Provider Identifies.";
    }
      description
        "Container for VPN Profiles.";
    }
    container vpn-services {
      list vpn-service {
        key "vpn-id";
        leaf vpn-id {
          type svc-id;
          description
            "Defining a service id.";

        }
        leaf vpn-svc-type {
          type identityref {
            base service-type;
          }
          default "vpws";
          description
            "Service type. By default, the service type is VPWS.";
        }
        leaf customer-name {
          type string;
          description
            "Customer name.";
        }
        leaf svc-topo {
          type identityref {
            base vpn-topology;
          }
          default "any-to-any";
          description
            "Defining service topology, such as
             any-to-any,hub-spoke, etc.";
```

```
      }
    container cloud-accesses {
      if-feature cloud-access;

      list cloud-access {
        key "cloud-identifier";
        leaf cloud-identifier {
          type leafref {
            path "/l2vpn-svc/vpn-profiles/valid-provider-identifiers"
              +"/cloud-identifier";
          }
          description
            "Identification of cloud service.
             Local administration meaning.";
        }
        choice list-flavor {
          case permit-any {
            leaf permit-any {
              type empty;
              description
                "Allow all sites.";
            }
          }
          case deny-any-except {
            leaf-list permit-site {
              type leafref {
                path "/l2vpn-svc/sites/site/site-id";
              }
              description
                "Site ID to be authorized.";
            }
          }
          case permit-any-except {
            leaf-list deny-site {
              type leafref {
                path "/l2vpn-svc/sites/site/site-id";
              }
              description
                "Site ID to be denied.";
            }
          }
          description
            "Choice for cloud access policy.
             By default, all sites in the L2VPN
             MUST be authorized to access the cloud.";
        }
        description
          "Cloud access configuration.";
```

```
      }
      description
        "Container for cloud access configurations";
    }
    container frame-delivery {
      if-feature bum;
      container customer-tree-flavors {
        leaf-list tree-flavor {
          type identityref {
            base multicast-tree-type;
          }
          description
            "Type of tree to be used.";
        }
        description
          "Type of trees used by customer.";
      }
    container bum-deliveries {
      list bum-delivery {
        key "frame-type";

        leaf frame-type {
          type identityref {
            base tf-type;
          }
          description
            "Type of frame delivery. It support unicast
             frame delivery, multicast frame delivery
              and broadcast frame delivery.";
        }
        leaf delivery-mode {
          type identityref {
            base frame-delivery-mode;
          }
          default "unconditional";
          description
            "Define Frame Delivery Mode
             (unconditional[default],
            conditional, or discard).
            By default, Service Frames
            are unconditionally delivered
            to the destination. ";
        }
        description
          "List of frame delivery type and mode.";
      }
      description
        "Define frame delivery type and mode.";
```

```
    }
      leaf multicast-gp-port-mapping {
        type identityref {
          base mapping-type;
        }
        mandatory true;
        description
          "Describe the way in which each interface is
           associated with the Multicast group";
      }
      description
        "Multicast global parameters for the VPN service.";
    }
    container extranet-vpns {
      if-feature extranet-vpn;
      list extranet-vpn {
        key "vpn-id";
        leaf vpn-id {
          type svc-id;
          description
            "Identifies the target VPN the local VPN want to access.";
        }
        leaf local-sites-role {
          type identityref {
            base site-role;
          }
          default "any-to-any-role";
          description
            "This describes the role of the local sites in the target
             VPN topology. In the any-to-any VPN service topology,
             the local sites must have the same role, which will be
             'any-to-any-role '. In the Hub-and-Spoke VPN service
             topology or the Hub and Spoke disjoint VPN service
             topology, the local sites must have a Hub role or a
              Spoke role";
        }
        description
          "List of extranet VPNs the local VPN is attached to.";
      }
      description
        "Container for extranet VPN configuration.";
    }
    leaf ce-vlan-preservation {
      type boolean;
      mandatory true;
      description
        "Preserve the CE-VLAN ID from ingress to egress,i.e.,
         CE-VLAN tag of the egress frame are identical to
```

```
          those of the ingress frame that yielded this
          egress service frame. If All-to-One bundling within
          a site is Enabled, then preservation applies to all
          Ingress service frames. If All-to-One bundling is
          Disabled , then preservation applies to tagged
          Ingress service frames having CE-VLAN ID 1
          through 4094.";
      }
      leaf ce-vlan-cos-perservation {
        type boolean;
        mandatory true;
        description
          "CE vlan CoS preservation. PCP bits in the CE-VLAN tag
           of the egress frame are identical to those of the
           ingress frame that yielded this egress service
           frame.";
      }
          leaf carrierscarrier {
            if-feature carrierscarrier;
            type boolean;
            default "false";
            description
              "The VPN is using CsC, and so MPLS
               is required.";
          }
          description
            "List of vpn services.";
        }
        description
          "Container for VPN services.";
      }
      container sites {
        list site {
          key "site-id";
          leaf site-id {
            type string;
            description
              "Identifier of the site.";
          }
      leaf site-vpn-flavor {
        type identityref {
          base site-vpn-flavor;
        }
        default "site-vpn-flavor-single";
        description
          "Defines the way the VPN multiplexing is
           done ,e.g.,whether the site belongs to
           a single VPN site or a multiVPN; By
```

```
         default, the site belongs to a single VPN.";
    }
    container devices {
      when "derived-from-or-self(../management/type, "
        +"'l2vpn-svc:provider-managed') or "
        +"derived-from-or-self(../management/type, "
        +"'l2vpn-svc:co-managed')" {
        description
          "Applicable only for provider-managed or
           co-managed device.";
      }
      list device {
        key "device-id";
        leaf device-id {
          type string;
          description
            "Identifier for the device.";
        }
        leaf location {
          type leafref {
            path "../../../locations/location/location-id";
          }
          mandatory true;
          description
            "Location of the device.";
        }
        container management {
          when "derived-from-or-self(../../../management/type,"
            +"'l2vpn-svc:co-managed')" {

            description
              "Applicable only for co-managed device.";
          }
          leaf transport {
            type identityref {
              base address-family;
            }
            description
              "Transport protocol or Address family
               used for management.";
          }
          leaf address {
            when "(../ transport)" {
              description
                "If address-family is specified, then address should
                 also be specified. If the transport is not specified,
                 then address should also not be specified.";
            }
```

```
            type inet:ip-address;
            description
              "Management address.";
          }
          description
            "Management configuration. Applicable only for
              co-managed device.";
        }
        description
          "List of devices requested by customer.";
      }
      description
        "Devices configuration";
    }
    container management {
      leaf type {
        type identityref {
          base management;
        }
        mandatory true;
        description
          "Management type of the connection.";
      }
      description

        "Management configuration.";
    }
    container locations {
      list location {
        key "location-id";
        leaf location-id {
          type string;
          description
            "Location ID";
        }
        leaf address {
          type string;
          description
            "Address (number and street) of the site.";
        }
        leaf postal-code {
          type string;
          description
            "postal code of the site. The format of postal-code is
             similar to postal code label format defined in
             RFC4119.";
        }
        leaf state {
```

```
          type string;
          description
            "State of the site. This leaf can also be used to
             describe a region for country who does not have
             states.";
        }
        leaf city {
          type string;
          description
            "City of the site.";
        }
        leaf country-code {
          type string;
          description
            "Country of the site. The format of country-code is similar
             to country label defined in RFC4119.";
        }
        description
          "List for location";
      }
      description
        "Location of the site.";
    }

    container site-diversity {
      if-feature site-diversity;
      container groups {
        list group {
          key "group-id";
          leaf group-id {
            type string;
            description
              "Group-id the site is belonging to";
          }
          description
            "List of group-id";
        }
        description
          "Groups the site is belonging to.
           All site network accesses will inherit those group
           values.";
      }
      description
        "Diversity constraint type.";
    }

    container vpn-policies {
      list vpn-policy {
```

```
        key "vpn-policy-id";
        leaf vpn-policy-id {
          type string;
          description
            "Unique identifier for the VPN policy.";
        }
        list entries {
          key "id";
          leaf id {
            type string;
            description
              "Unique identifier for the policy entry.";
          }
          container filters {
            list filter {
              key "type";
              ordered-by user;
              leaf type {
                type identityref {
                  base vpn-policy-filter-type;
                }
                description
                  "Type of VPN Policy filter.";
              }
              leaf-list lan-tag {
                when "derived-from-or-self(../type, 'l2vpn-svc:lan')" {
                  description
                    "Only applies when VPN Policy filter is
                     LAN Tag filter.";
                }
                if-feature lan-tag;
                type uint32;
                description
                  "List of Ethernet LAN Tag to be matched. Ethernet
                   LAN Tag identifies a particular broadcast domain
                   in a VPN. ";
              }
              description
                "List of filters used on the site. This list can
                 be augmented.";
            }
            description
              "If a more-granular VPN attachment is necessary,
               filtering can be used. If used, it permits the
               splitting of site LANs among multiple VPNs. The
               Site LAN can be split based on either LAN-tag or
               LAN prefix. If no filter is used, all the LANs
               will be part of the same VPNs with the same
```

```
              role.";
          }
          list vpn {
            key "vpn-id";
            leaf vpn-id {
              type leafref {
                path "/l2vpn-svc/vpn-services/vpn-service/vpn-id";
              }
              description
                "Reference to L2VPN.";
            }
            leaf site-role {
              type identityref {
                base site-role;
              }
              default "any-to-any-role";
              description
                "Role of the site in the L2VPN.";
            }
            description
              "List of VPNs the LAN is associated with.";
          }
          description
            "List of entries for export policy.";
        }
        description
          "List of VPN policies.";
      }
      description
        "VPN policy.";
    }
    container service {
      uses site-service-qos-profile;
      uses site-service-mpls;
      description
        "Service parameters on the attachment.";
    }
        uses site-bum;
        uses site-mac-loop-prevention;
        uses site-acl;
    leaf actual-site-start {
      type yang:date-and-time;
      config false;
      description
        "Optional leaf indicating actual date
         and time when the service at a particular
         site actually started";
    }
```

```
    leaf actual-site-stop {
      type yang:date-and-time;
      config false;
      description
        "Optional leaf indicating actual date
         and time when the service at a particular
         site actually stopped";
    }
    leaf bundling-type {
      type identityref {
        base bundling-type;
      }
      default "one2one-bundling";
      description
        "Bundling type. By default, Each L2VPN
         can be associated with only one
         CE-VLAN ,i.e., one to one bundling is used.";
    }
    leaf default-ce-vlan-id {
      type uint32;
      mandatory true;
      description
        "Default CE VLAN ID set at site level.";
    }
        container site-network-accesses {
          list site-network-access {
            key "network-access-id";
            leaf network-access-id {
              type string;
              description
                "Identifier of network access";
            }
            leaf remote-carrier-name {
              when "derived-from-or-self(../../../site-vpn-flavor,"+
                "'l2vpn-svc:site-vpn-flavor-nni')" {
                description
                  "Relevant when Site vpn flavor is
                   site-vpn-flavor-nni.";
              }
              type leafref {
                path "/l2vpn-svc/vpn-profiles/"+
                 "valid-provider-identifiers"+
                  "/remote-carrier-identifier";
              }
              description
                "Remote carrier name. The remote-carrier-name
                 must be configured only when site-vpn-flavor
                 is set to site vpn-flavor-nni. If it is not
```

```
                set,it indicates customer does not know remote
                carrier name beforehand.";
            }
        leaf type {
          type identityref {
            base site-network-access-type;
          }
          default "point-to-point";
          description
            "Describes the type of connection, e.g.,
             point-to-point or multipoint.";
        }
        choice location-flavor {
          case location {
            when "derived-from-or-self(../../management/type, "
              +"'l2vpn-svc:customer-managed')" {
              description
                "Applicable only for customer-managed device.";
            }
            leaf location-reference {
              type leafref {
                path "../../../locations/location/location-id";
              }
              description
                "Location of the site-network-access.";
            }
          }
          case device {
            when "derived-from-or-self(../../management/type, "
              +"'l2vpn-svc:provider-managed') or "
              +"derived-from-or-self(../../management/type, "
              +"'l2vpn-svc:co-managed')" {
              description
                "Applicable only for provider-managed
                 or co-managed device.";
            }
            leaf device-reference {
              type leafref {
                path "../../../devices/device/device-id";
              }
              description
                "Identifier of CE to use.";
            }
          }
          mandatory true;
          description
            "Choice of how to describe the site's location.";
        }
```

```
    container access-diversity {
      if-feature site-diversity;
    container groups {
      list group {
        key "group-id";
        leaf group-id {
          type string;
          description
            "Group-id the site is belonging to.";
        }
        description
          "List of group-id";
      }
      description
        "Groups the site or site-network-access
         is belonging to.";
    }
      container constraints {
        list constraint {
          key "constraint-type";
          leaf constraint-type {
            type identityref {
              base placement-diversity;
            }
            description
              "Diversity constraint type.";
          }
          container target {
            choice target-flavor {
              default "id";
              case id {
                list group {
                  key "group-id";
                  leaf group-id {
                    type string;
                    description
                      "The constraint will apply
                       against this particular
                       group-id.";
                  }
                  description
                    "List of groups.";
                }
              }
              case all-accesses {
                leaf all-other-accesses {
                  type empty;
                  description
```

```
                    "The constraint will apply
                     against all other site network
                     access of this site.";
                }
              }
              case all-groups {

                leaf all-other-groups {
                  type empty;
                  description
                    "The constraint will apply
                     against all other groups the
                     customer is managing.";
                }
              }
              description
                "Choice for the group definition.";
            }
            description
              "The constraint will apply against
               this list of groups.";
          }
          description
            "List of constraints.";
        }
        description
          "Constraints for placing this site
           network access.";
      }
      description
        "Diversity parameters.";
    }
    container bearer {
      container requested-type {
        if-feature requested-type;
        leaf type {
          type string;
          description
            "Type of requested bearer Ethernet, ATM, Frame
             Relay, IP Layer 2 Transport, Frame Relay DLCI,
             SONET/SDH,PPP.";
        }
        leaf strict {
          type boolean;
          default "false";
          description
            "Define if the requested-type is a preference
             or a strict requirement.";
```

```
        }
        description
          "Container for requested type.";
      }
      leaf always-on {
        if-feature always-on;
        type boolean;
        default "true";
        description
          "Request for an always on access type.
           For example.This could mean no Dial access type.";
      }
      leaf bearer-reference {
        if-feature bearer-reference;
        type string;
        description
          "This is an internal reference for the
           service provider.";
      }
      description
        "Bearer specific parameters.
         To be augmented.";
    }
    container connection {
      leaf encapsulation-type {
        type identityref {
          base encapsulation-type;
        }
        default "ethernet";
        description
          "Encapsulation Type. By default,the
           encapsulation type is set as Ethernet.";
      }
      leaf eth-inf-type {
        type identityref {
          base eth-inf-type;
        }
        default "untagged";
        description
          "Ethernet Interface Type. By default,
          the Ethernet Interface Type is set as
          untagged interface.";
      }
    container tagged-interface {
      leaf type {
        type identityref {
          base tagged-inf-type;
        }
```

```
        default "priority-tagged";
        description
          "Tagged interface type. By default,
           the Tagged interface type is priority
           tagged interface. ";
      }
      container dot1q-vlan-tagged {
        when "derived-from-or-self(../type, 'l2vpn-svc:dot1q')" {
          description
            "Only applies when Tagged interface type is dot1q.";
        }
        if-feature dot1q;
        leaf tg-type {
          type identityref {
            base tag-type;
          }
          default "c-vlan";
          description
            "TAG type.By default, Tag type is Customer-VLAN tag.";
        }
        leaf cvlan-id {
          type uint16;
          mandatory true;
          description
            "VLAN identifier.";
        }
        description
          "Tagged interface.";
      }
      container priority-tagged {
        when "derived-from-or-self(../type, "
          +"'l2vpn-svc:priority-tagged')" {
          description
            "Only applies when Tagged interface type
             is priority tagged interface.";
        }
        leaf tag-type {
          type identityref {
            base tag-type;
          }
          default "c-vlan";
          description
            "TAG type.By default, the TAG type is
             Customer-VLAN tag.";
        }
        description
          "Priority tagged.";
      }
```

```
      container qinq {
        when "derived-from-or-self(../type, 'l2vpn-svc:qinq')" {
          description
            "Only applies when Tagged interface type is qinq.";
        }
        if-feature qinq;
        leaf tag-type {
          type identityref {
            base tag-type;
          }
          default "c-s-vlan";
          description
            "Tag type. By default, the Tag type is c-s-vlan.";
        }
        leaf svlan-id {
          type uint16;
          mandatory true;
          description
            "S-VLAN Identifier.";
        }
        leaf cvlan-id {
          type uint16;
          mandatory true;
          description
            "C-VLAN Identifier";
        }
        description
          "QinQ.";
      }
      container qinany {
        when "derived-from-or-self(../type, 'l2vpn-svc:qinany')" {
          description
            "Only applies when Tagged interface type is qinany.";
        }
        if-feature qinany;
        leaf tag-type {
          type identityref {
            base tag-type;
          }
          default "s-vlan";
          description
            "Tag type.By default, the Tag type is Service-VLAN tag.";
        }
        leaf svlan-id {
          type uint16;
          mandatory true;
          description
            "S-Vlan ID.";
```

```
        }
        description
          "Container for Qin Any.";
      }
      container vxlan {
        when "derived-from-or-self(../type, 'l2vpn-svc:vxlan')" {
          description
            "Only applies when Tagged interface type is vxlan.";
        }
        if-feature vxlan;
        leaf vni-id {
          type uint32;
          mandatory true;
          description
            "VNI Identifier.";
        }
        leaf peer-mode {
          type identityref {
            base vxlan-peer-mode;
          }
          default "static-mode";
          description
            "specify the vxlan access mode. By default
             the peer mode is Set as static mode.";
        }
        list peer-list {
          key "peer-ip";
          leaf peer-ip {
            type inet:ip-address;
            description
              "Peer IP.";
          }
          description
            "List for peer IP.";
        }
        description
          "QinQ.";
      }

      description
        "Container for tagged Interface.";
    }
    container untagged-interface {
      leaf speed {
        type uint32;
        units "mbps";
        default "10";
        description
```

```
          "Port speed.";
      }
      leaf mode {
        type neg-mode;
        default "auto-neg";
        description
          "Negotiation mode.";
      }
      leaf phy-mtu {
        type uint32;
        units "bytes";
        description
          "PHY MTU.";
      }
      leaf lldp {
        type boolean;
        default "false";
        description
          "LLDP. Indicate LLDP is supported.";
      }
      container oam-802.3ah-link {
        if-feature oam-3ah;
        leaf enable {
          type boolean;
          default "false";
          description
            "Indicate whether support oam 802.3 ah link";
        }
        description
          "Container for oam 802.3 ah link.";
      }
      leaf uni-loop-prevention {
        type boolean;
        default "false";
        description
          "If this leaf set to truth that the port automatically
           goes down when a physical loopback is detect.";
      }
      description
        "Container of Untagged Interface Attributes
         configurations.";
    }
    container lag-interfaces {
      if-feature lag-interface;
      list lag-interface {
        key "index";
        leaf index {
          type string;
```

```
          description
            "LAG interface index.";
        }
    container lacp {
      if-feature lacp;
      leaf enable {
        type boolean;

        default "false";
        description
          "LACP on/off.By default, LACP is disabled.";
      }
      leaf mode {
        type neg-mode;
        description
          "LACP mode. LACP modes have auto negotiation mode
           and passive mode (false). Auto negotiation mode
           means initiating the auto speed negotiation and
           trying to form Ethernet Channel with other end.
           Passive mode means not initiating the negotiation,
          but responding to LACP packets initiated by other
          end(e.g., full duplex or half duplex. ";
      }
      leaf speed {
        type uint32;
        units "mbps";
        default "10";
        description
          "LACP speed. By default, the lacp speed is 10Mbps.";
      }
      leaf mini-link-num {
        type uint32;
        description
          "Defines the minimum number of links that must be
           active before the aggregating link is put
           into service.";
      }
      leaf system-priority {
        type uint16;
        default "32768";
        description
          "Indicates the LACP priority for the system.
           The range is from 0 to 65535.
           The default is 32768.";
      }
      container micro-bfd {
        if-feature micro-bfd;
        leaf enable {
```

```
          type enumeration {
            enum "on" {
              description
                "Micro-bfd on.";
            }
            enum "off" {
              description
                "Micro-bfd off.";
            }
          }
          default "off";
          description
            "Micro BFD ON/OFF. By default,
             the micro-bfd is set to off.";
        }
        leaf interval {
          type uint32;
          units "msec";
          description
            "BFD interval.";
        }
        leaf hold-timer {
          type uint32;
          units "msec";
          description
            "BFD hold timer.";
        }
        description
          "Container of Micro-BFD configurations.";
      }
      container bfd {
        if-feature bfd;
        leaf enabled {
          type boolean;
          default "false";
          description
            "BFD activation. By default, BFD is not activated.";
        }
        choice holdtime {
          default "fixed";
          case profile {
            leaf profile-name {
              type leafref {
                path "/l2vpn-svc/vpn-profiles/"
                 +"valid-provider-identifiers"
                 +"/bfd-profile-identifier";
              }
              description
```

```
              "Service provider well known profile.";
            }
            description
              "Service provider well known profile.";
          }
          case fixed {
            leaf fixed-value {
              type uint32;

              units "msec";
              description
                "Expected hold time expressed in msec.";
            }
          }
          description
            "Choice for hold time flavor.";
        }
        description
          "Container for BFD.";
      }
      container member-links {
        list member-link {
          key "name";
          leaf name {
            type string;
            description
              "Member link name.";
          }
          leaf speed {
            type uint32;
            units "mbps";
            default "10";
            description
              "Port speed.";
          }
          leaf mode {
            type neg-mode;
            default "auto-neg";
            description
              "Negotiation mode.";
          }
          leaf link-mtu {
            type uint32;
            units "bytes";
            description
              "Link MTU size.";
          }
          container oam-802.3ah-link {
```

```
            if-feature oam-3ah;
            leaf enable {
              type boolean;
              default "false";
              description
                "Indicate whether oam 802.3 ah link is supported.";
            }
            description
              "Container for oam 802.3 ah link.";
          }
          description
            "Member link";
        }
        description
          "Container of Member link list";
      }
      leaf flow-control {
        type boolean;
        default "false";
        description
          "Flow control. Indicate whether flow control is supported.";
      }
      leaf lldp {
        type boolean;
        default "false";
        description
          "LLDP. Indicate whether lldp is supported.";
      }
      description
        "LACP.";
    }
      description
        "List of LAG interfaces.";
    }
    description
      "Container of LAG interface attributes configuration";
  }
  list cvlan-id-to-svc-map {
    key "svc-id";
    leaf svc-id {
      type leafref {
        path "/l2vpn-svc/vpn-services/vpn-service/vpn-id";
      }
      description
        "VPN Service identifier";
    }
    list cvlan-id {
      key "vid";
```

```
        leaf vid {
          type uint16;
          description
            "CVLAN ID";

        }
        description
          "List of CVLAN-ID to SVC Map configurations";
      }
      description
        "List for cvlan-id to L2VPn Service map configurations";
    }
    container l2cp-control {
      if-feature l2cp-control;
      leaf stp-rstp-mstp {
        type control-mode;
        description
          "STP/RSTP/MSTP protocol type applicable to all Sites.";
      }
      leaf pause {
        type control-mode;
        description
          "Pause protocol type applicable to all Sites.";
      }
      leaf lacp-lamp {
        type control-mode;
        description
          "LACP/LAMP.";
      }
      leaf link-oam {
        type control-mode;
        description
          "Link OAM.";
      }
      leaf esmc {
        type control-mode;
        description
          "ESMC.";
      }
      leaf l2cp-802.1x {
        type control-mode;

        description
          "IEEE 802.x.";
      }
      leaf e-lmi {
        type control-mode;
        description
```

```
          "E-LMI.";
      }
      leaf lldp {
        type boolean;
        description
          "LLDP protocol type applicable to all sites.";
      }
      leaf ptp-peer-delay {
        type control-mode;
        description
          "PTP peer delay.";
      }
      leaf garp-mrp {
        type control-mode;
        description
          "GARP/MRP.";
      }
      description
        "Container of L2CP control configurations";
    }
    container oam {
      if-feature ethernet-oam;
      leaf md-name {
        type string;
        mandatory true;
        description
          "Maintenance domain name.";
      }
      leaf md-level {
        type uint16 {
          range "0..255";
        }
        mandatory true;
        description
          "Maintenance domain level. The level may be
           restricted in certain protocols (e.g.,
           protocol in layer 0 to layer 7).";
      }
  list cfm-8021-ag {
    if-feature cfm;
    key "maid";

    leaf maid {
      type string;
      mandatory true;
      description
        "Identify an Maintenance Association (MA).";
    }
```

```
    leaf mep-id {
      type uint32;
      description
        "Local Maintenance End Point (MEP) ID.
         The non-existence of this leaf means
          that no defects are to be reported.";
    }
    leaf mep-level {
      type uint32;
      description
        "Define Maintenance End Point (MEP) level.
         The non-existence of this leaf means that
         no defects are to be reported.";
    }
    leaf mep-up-down {
      type enumeration {
        enum "up" {
          description
            "MEP up.";
        }
        enum "down" {
          description
            "MEP down.";
        }
      }
      default "up";
      description
        "MEP up/down. By default, MEP up is used.
         The non-existence of this leaf means that
         no defects are to be reported.";
    }
    leaf remote-mep-id {
      type uint32;
      description
        "Remote MEP ID. The non-existence of this leaf means
         that no defects are to be reported.";
    }
    leaf cos-for-cfm-pdus {
      type uint32;
      description
        "COS for CFM PDUs. The non-existence of this leaf means
         that no defects are to be reported.";

    }
    leaf ccm-interval {
      type uint32;
      units "msec";
      default "10000";
```

```
      description
        "Continuity Check Message(CCM) interval.
         By default, ccm-interval is 10 seconds.";
    }
    leaf ccm-holdtime {
      type uint32;
      units "msec";
      default "35000";
      description
        "CCM hold time. By default ccm hold time
         is 3.5 times of ccm interval.";
    }
    leaf alarm-priority-defect {
      type identityref {
        base fault-alarm-defect-type;
      }
      default "remote-invalid-ccm";
      description
        "The lowest priority defect that is
         allowed to generate a Fault Alarm.By default,
         fault-alarm-defect-type is set to remote-invalid-ccm.
          The non-existence of this leaf means
          that no defects are to be reported.";
    }
    leaf ccm-p-bits-pri {
      type ccm-priority-type;
      description
        "The priority parameter for CCMs transmitted by the MEP.
         The non-existence of this leaf means
         that no defects are to be reported.";
    }
     description
       "List of 802.1ag CFM attributes";
    }
    list y-1731 {
      if-feature y-1731;
      key "maid";
      leaf maid {

        type string;
        mandatory true;
        description
          "Identify an Maintenance Association (MA).";
      }
      leaf mep-id {
        type uint32;
        description
          "Local Maintenance End Point(MEP) ID.
```

```
           The non-existence of this leaf means
           that no measurements are to be reported.";
        }
        leaf type {
          type identityref {
            base pm-type;
          }
          default "delay";
          description
            "Performance monitor types. By default, the
             performance monitoring type is set to delay.
             The non-existence of this leaf means that no
              measurements are to be reported.";
        }
        leaf remote-mep-id {
          type uint32;
          description
            "Remote MEP ID. The non-existence of this
             leaf means that no measurements are to be
             reported.";
        }
        leaf message-period {
          type uint32;
          units "msec";
          default "10000";
          description
            "Defines the interval between Y.1731
             performance monitoring messages. The message
             period is expressed in milliseconds.";
        }
        leaf measurement-interval {
          type uint32;
          units "sec";
          description
            "Specifies the measurement interval for statistics. The
             measurement interval is expressed in seconds.";
        }
        leaf cos {
          type uint32;

          description
            "Class of service. The non-existence
             of this leaf means that no measurements
             are to be reported.";
        }
        leaf loss-measurement {
          type boolean;
          default "false";
```

```
        description
          "Whether enable loss measurement.
           By default, loss measurement is not
           enabled.";
      }
      leaf synthethic-loss-measurement {
        type boolean;
        default "false";
        description
          "Indicate whether enable synthetic loss
           measurement.By default, synthethic loss
           measurement is not enabled.";
      }
      container delay-measurement {
        leaf enable-dm {
          type boolean;
          default "false";
          description
            "Whether to enable delay measurement.
            By defaut, the delay measurement is
             not enabled.";
        }
        leaf two-way {
          type boolean;
          default "false";
          description
            "Whether delay measurement is two-way (true) of one-
             way (false). By default, one way measurement is enabled.";
        }
        description
          "Container for delay measurement.";
      }
      leaf frame-size {
        type uint32;
        units "bytes";
        description
          "Frame size. The non-existence of this leaf means
          that no measurements are to be reported.";
      }
      leaf session-type {
        type enumeration {
          enum "proactive" {
            description
              "Proactive mode.";
          }
          enum "on-demand" {
            description
              "On demand mode.";
```

```
          }
        }
        default "on-demand";
        description
          "Session type. By default, the session type is on demand mode.
           The non-existence of this leaf means that no measurements
           are to be reported.";
      }
      description
        "List for y-1731.";
    }
      description
        "Container for Ethernet service OAM.";
    }
      description
        "Container for bearer";
    }
    container availability {
      leaf access-priority {
        type uint32;
        default "100";
        description
          "Access priority. The higher the access-priority value,
           the higher the preference of the access will be.";
      }
      choice redundancy-mode {
        case single-active {
          leaf single-active {
            type empty;
            description
              "Single active.";
          }
          description
            "Single active case.";
        }
        case all-active {

          leaf all-active {
            type empty;
            description
              "All active.";
          }
          description
            "All active case.";
        }
        description
          "Redundancy mode choice.";
      }
```

```
      description
        "Container of availability optional configurations.";
    }
    container vpn-attachment {
      choice attachment-flavor {
        case vpn-id {
          leaf vpn-id {
            type leafref {
              path "/l2vpn-svc/vpn-services/vpn-service/vpn-id";
            }
            description
              "Reference to a L2VPN. Referencing a vpn-id provides
               an easy way to attach a particular logical access to
               a VPN. In this case, vpn-id must be configured.";
          }
          leaf site-role {
            type identityref {
              base site-role;
            }
            default "any-to-any-role";
            description
              "Role of the site in the L2VPN. When referencing a vpn-id,
               the site-role setting must be added to express the role
               of the site in the target VPN service topology.";
          }
        }
        case vpn-policy-id {
          leaf vpn-policy-id {
            type leafref {
              path "../../../../vpn-policies/vpn-policy/vpn-policy-id";
            }
            description
              "Reference to a vpn policy.";
          }
        }
        mandatory true;
        description
          "Choice for VPN attachment flavor.";
      }
      description
        "Defines VPN attachment of a site.";
    }
    container service {
     container svc-bandwidth {
      if-feature input-bw;
      list bandwidth {
        key "direction type";
        leaf direction {
```

```
          type identityref {
            base bw-direction;
          }
          description
            "Indicate the bandwidth direction. It can be bandwidth
             download direction from the SP to the site or
             bandwidth upload direction from the site to the SP.";
        }
        leaf type {
          type identityref {
            base bw-type;
          }
          description
            "Bandwidth Type. By default, the bandwidth type is set
             as bandwidth per cos.";
        }
        leaf cos-id {
                  when "derived-from-or-self(../type, 'l2vpn-svc:bw-per-cos')" {
            description

              "Relevant when the bandwidth type is set as bandwidth
               per cos.";
          }
          type uint8;
          description
            "Identifier of Class of Service
             , indicated by DSCP or a CE-CLAN
             CoS(802.1p)value in the service frame.
            If bandwidth type is set as bandwidth
            per cos, cos-id MUST be specified.";
        }
        leaf vpn-id {
                  when "derived-from-or-self(../type, "
                    +"'l2vpn-svc:bw-per-svc')" {
            description
              "Relevant when the bandwidth type is
               set as bandwidth per VPN service.";
          }
          type svc-id;
          description
            "Identifies the target VPN. If bandwidth
             type is set as bandwidth per VPN service,
             vpn-id MUST be specified.";
        }
        leaf cir {
          type uint64;
          units "bps";
          mandatory true;
```

```
          description
            "Committed Information Rate. The maximum number of
             bits that a port can receive or send during
             one-second over an interface.";
        }
        leaf cbs {
          type uint64;
          units "bps";
          mandatory true;
          description
            "Committed Burst Size.CBS controls the bursty nature
             of the traffic. Traffic that does not use the configured
             CIR accumulates credits until the credits reach the
             configured CBS.";
        }
        leaf eir {
          type uint64;
          units "bps";
          description
            "Excess Information Rate,i.e.,Excess frame delivery
             allowed not subject to SLA.The traffic rate can be
             limited by eir.";
        }
        leaf ebs {
          type uint64;
          units "bps";
          description
            "Excess Burst Size. The bandwidth available for burst
             traffic from the EBS is subject to the amount of
             bandwidth that is accumulated during periods when
             traffic allocated by the EIR policy is not used.";
        }
        leaf pir {
          type uint64;
          units "bps";
          description
            "Peak Information Rate, i.e., maixmum frame delivery
             allowed.It is equal to or less than sum of cir
             and eir.";
        }
        leaf pbs {
          type uint64;
          units "bps";
          description
            "Peak Burst Size. It is measured in bytes per second.";
        }
        description
          "List for bandwidth.";
```

```
      }
      description
        "From the customer site's perspective, the service
         input/out bandwidth of the connection or download/upload
         bandwidth from the SP/site to the site/SP.";
    }
    leaf svc-mtu {
      type uint16;
      units "bytes";
      mandatory true;
      description
        "SVC MTU, it is also known as the maximum transmission unit or
         maximum frame size,When a frame is larger than the MTU, it is
         broken down, or fragmented, into smaller pieces by the network
         protocol to accommodate the MTU of the network. If CsC is
         enabled,the requested svc-mtu leaf will refer to the
         MPLS MTU and not to the link MTU. ";
    }
      uses site-service-qos-profile;
      uses site-service-mpls;
      description
        "Container for service";
    }
    uses site-bum;
    uses site-mac-loop-prevention;
    uses site-acl;
    container mac-addr-limit {
      if-feature mac-addr-limit;
      leaf limit-number {
        type uint16;
        default "2";
        description
          "maximum number of MAC addresses learned from
           the subscriber for a single service instance.
           The default allowed maximum number of MAC

           addresses is 2.";
      }
      leaf time-interval {
        type uint32;
        units "sec";
        default "300";
        description
          "The aging time of the mac address. By default,
           the aging time is set 300 seconds.";
      }
      leaf action {
        type identityref {
```

```
          base mac-action;
        }
        default "warning";
        description
          "specify the action when the upper limit is
           exceeded: drop the packet, flood the
           packet, or simply send a warning log message. By default,
           action is set to warning.";
      }
      description
        "Container of MAC-Addr limit configurations";
    }
           description
             "List of Site Network Accesses.";
         }
         description
           "Container of port configurations.";
       }
       description
         "List of sites.";

     }
     description
       "Container of site configurations.";
   }
   description
     "Container for L2VPN service.";
 }
}
<CODE ENDS>
```

9. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF[RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH)[RFC6242] . The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the

default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

- /l2vpn-svc/vpn-services/vpn-service

  The entries in the list above include the whole vpn service configurations which the customer subscribes, and indirectly create or modify the PE and CE device configurations. Unexpected changes to these entries could lead to the service disruption and/or network misbehavior.

- /l2vpn-svc/sites/site

  The entries in the list above include the customer site configurations. As above, unexpected changes to these entries could lead to the service disruption and/or network misbehavior.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

- /l2vpn-svc/vpn-services/vpn-service

- /l2vpn-svc/sites/site

The entries in the lists above include customer-proprietary or confidential information, e.g., customer-name, site location, what service the customer subscribes.

When a Service Provider collaborates with multiple customers, it have to ensure that a given customer can only view and modify his own service information.

The data model defines some security parameters that can be extended via augmentation as part of the customer service request; those parameters are described in Section 5.12 and Section 5.13.

10. IANA Considerations

IANA is requested to assign a new URI from the IETF XML registry ([RFC3688]). The following URI is suggested:

URI: urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace

This document also requests a new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion:

name: ietf-l2vpn-svc
namespace: urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc
prefix: l2vpn-svc
reference: RFC XXXX

11. Acknowledgements

Thanks to Qin Wu and Adrian Farrel for facilitating work on the initial revisions of this document. Thanks to Zonghe Huang, Wei Deng and Xiaoling Song to help review this draft.

Special thanks to Jan Lindblat for his careful review of the YANG.

This document has drawn on the work of the L3SM Working Group expressed in [RFC8299].

12. References

12.1. 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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[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>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[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>.

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8214] Boutros, S., Sajassi, A., Salam, S., Drake, J., and J. Rabadan, "Virtual Private Wire Service Support in Ethernet VPN", RFC 8214, DOI 10.17487/RFC8214, August 2017, <https://www.rfc-editor.org/info/rfc8214>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

12.2. Informative References

[I-D.ietf-bess-evpn-yang]
Brissette, P., Shah, H., Hussain, I., Tiruveedhula, K., and J. Rabadan, "Yang Data Model for EVPN", draft-ietf-bess-evpn-yang-05 (work in progress), February 2018.

[I-D.ietf-bess-l2vpn-yang]
Shah, H., Brissette, P., Chen, I., Hussain, I., Wen, B., and K. Tiruveedhula, "YANG Data Model for MPLS-based L2VPN", draft-ietf-bess-l2vpn-yang-08 (work in progress), February 2018.

[IEEE-802-1ag]
IEEE, "802.1ag - Connectivity Fault Management", December 2007.

[IEEE-802-1D]
IEEE, "802.1D-2004 - MAC Bridges", June 2004.

[ITU-T-Y-1731]
ITU-T, "Recommendation Y.1731 - OAM functions and mechanisms for Ethernet based networks", February 2008.

[MEF-6] MEF Forum, "Ethernet Services Definitions - Phase 2", April 2008.

[RFC4119] Peterson, J., "A Presence-based GEOPRIV Location Object Format", RFC 4119, DOI 10.17487/RFC4119, December 2005, <https://www.rfc-editor.org/info/rfc4119>.

[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>.

[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>.

[RFC7130] Bhatia, M., Ed., Chen, M., Ed., Boutros, S., Ed., Binderberger, M., Ed., and J. Haas, Ed., "Bidirectional Forwarding Detection (BFD) on Link Aggregation Group (LAG) Interfaces", RFC 7130, DOI 10.17487/RFC7130, February 2014, <https://www.rfc-editor.org/info/rfc7130>.

[RFC7209] Sajassi, A., Aggarwal, R., Uttaro, J., Bitar, N., Henderickx, W., and A. Isaac, "Requirements for Ethernet VPN (EVPN)", RFC 7209, DOI 10.17487/RFC7209, May 2014, <https://www.rfc-editor.org/info/rfc7209>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[RFC7436] Shah, H., Rosen, E., Le Faucheur, F., and G. Heron, "IP-Only LAN Service (IPLS)", RFC 7436, DOI 10.17487/RFC7436, January 2015, <https://www.rfc-editor.org/info/rfc7436>.

[RFC8199] Bogdanovic, D., Claise, B., and C. Moberg, "YANG Module Classification", RFC 8199, DOI 10.17487/RFC8199, July 2017, <https://www.rfc-editor.org/info/rfc8199>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

Appendix A. Changes Log

Changes in v-(01) include:

- Reference Update.
- Fix figure in section 3.3 and section 3.4
- Consider VPWS, VPLS, EVPN as basic service and view EVC related service as additional service.
- Model structure change, move two customer information related parameter into VPN Services container, remove ’customer-info ’container
- Redefine vpn-type to cover VPWS, VPLS, EVPN service;
- Consolidate EVC and OVC container, make them optional since for some L2VPN service such as EVPN sevice, OVC, EVC are not needed.
- Add service and security filter under sites container and change "ports" into "site-network-accesses" to get consistent with L3SM and also make it generalized.
- Fixed usage examples in the l2sm model draft.

Changes in v-(02) include:

- Fix figure 3 and figure 4 in section 3.4 to apply IEEE802.3 on the segment between C and CE and apply IEEE802.1Q on the segment between CE and PE.
- Update Signaling Option section and add L2TP support and classify the signaling option type into BGP-L2VPN, BGP-EVPN, LDP-PWE, L2TP-PW.
- Add Multicast Support in section 5.2.13, section 5.10.3 and move the text in BUM Storm Control section into section 5.10.3.
- Add new section 5.3.1, section 5.4, section 5.5, section 5.6, section 5.7, section 5.8, section 5.11to explain the usage of constraint parameters and service placement related parameters.
- Add new section 5.1 and 5.14 to allow augmentation and external ID References.
- Add new section to discuss inter-AS support and inter-provider support with NNI and EVC, OVC.

- Update Service Section 5.10 and define four type for svc-input-bandwidth and svc-output-bandwidth and add guaranteed-bw-percent parameter and related description.
- Add extranet VPN support.
- Remove duplicated parameters from cloud access.
- Move L2CP control plane protocol parameters under connection.
- Update section 5.3.3.2 to address loop avoidance issue and divide section 5.3.3.2 into Physical interface section, LAG interface section and Addressing Section.
- Reference Update.

Changes in v-(03) include:

- Introduce additional terminology.
- Modify figure 5 to get consistent with RFC8049.
- Add end to end Multi-segment connectivity support and site-vpn-flavor-e2e attribute.
- Add usage example to explain how to use EVC and OVC.
- Discuss applicability of this model to inter-provider support.
- Reduce redundant parameters related to encapsulation type and Ethernet type in the model.
- Clarify the relationship between guarantee-bandwidth-percent and CIR, EIR and PIR.
- Modify model structure for VPN service to make it consistent with the text in section 5.
- Remove Sub-inf parameter since it is similar to QinQ parameter.
- Add "direction" parameter for QoS profile.
- Update XML example and figure in section 5.16.

Changes in v-(04) include:

- Remove EVC and OVC related attributes.

- Remove Metro-Network related attributes.
- Remove Customer Account Number attributes.
- Update L2VPN service Types.
- Remove load banlancing options since access-priority within availability can be used to support load balancing.
- Remove service protection attribute since we have site diversity attributes.
- Move SVC-MTU to service level.
- Move CVLAN to Service Mapping to Network Access Level.
- Add two new parameters under qos-classification-policy.
- Remove Security Container.
- Remove IPv4/IPv6 prefix filter from VPN policy.
- Add Delivery mode support at service level.

Changes in v-(05) include:

- Change type from 16-bit integer to string for the leaf id under "qos-classification-policy" container.
- Stick to using ordered-by user and remove inefficiency to map service model sequence number to device model sequence number.
- Remove mandating the use of deviations and add "if-feature target-sites" under the leaf-list target-sites in section 5.10.2.
- RFC2119 language changes on operation of the management system in Section 5.6,3rd paragraph and section 7.
- Fix incomplete description statements.
- Change the use of the absolute paths to the use of relative paths in the "must" statement or "path" statement for vpn-policy-id leaf node, management container, location leaf node, devices container, location case, location-reference leaf, device case, device-reference leaf to make configuration is only applicable to the current sites.

- Change "must" statement to "when" statement for management container device container.
- Define new grouping vpn-profile-cfg for all the identifiers provided by SP to the customer. The identifiers include cloud-identifier, std-qos-profile.
- Add in the XPATH string representation and remove unqualified name.
- Remove redundant parameters in the cloud access.
- Add a few text to clarify what the site is in section 6.3.
- Add multi-filter and multi-VPN per entry support for VPN policy.
- Modify description for svc-bandwidth leaf to make it consistent with the text in section 5.10.1.
- Add text to clarify the way to achieve Per-VPN QoS policy.
- Change guaranteed-bw-percent data type from uint8 to decimal64.

Authors' Addresses

Bin Wen
Comcast

Email: bin_wen@comcast.com

Giuseppe Fioccola (editor)
Telecom Italia

Email: giuseppe.fioccola@telecomitalia.it

Chongfeng Xie
China Telecom

Email: xiechf@ctbri.com.cn

Luay Jalil
Verizon

Email: luay.jalil@verizon.com