L2SM Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 22, 2018

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

September 18, 2017

A YANG Data Model for L2VPN Service Delivery
draft-ietf-l2sm-l2vpn-service-model-03

Abstract

This document defines a YANG data model that can be used to configure a Layer 2 Provider Provisioned VPN service.

This model is intended to be instantiated at management system to deliver the overall service. This model is not a configuration model to be used directly on network elements, but provides an abstracted view of the Layer 2 VPN service configuration components. 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 data 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.

Requirements Language

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

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 March 22, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document defines a YANG data model for Layer 2 VPN (L2VPN) service configuration. This model is intended to be instantiated at management system to allow a user (a customer or an application) to request the service from a service provider. This model is not a configuration model to be used directly on network elements, but provides an abstracted view of the L2VPN service configuration components. 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 data 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].

Further discussion of the way that services are modeled in YANG and of the relationship between "customer service models" like the one described in this document and configuration models can be found in [I-D.ietf-opsawg-service-model-explained]. 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.

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 [RFC6020] and are not redefined here:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC6020].

1.2. Tree diagram

A simplified graphical representation of the data model is presented in Section 5.

The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node and "*" denotes a "list" or "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

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 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 VSI.

UNI: The physical demarcation point between the responsibility of Customer and the responsibility of Provider.

ENNI: a reference point representing the boundary between two Operator Networks that are operated as separate administrative domains.

Ethernet Virtual Connection(EVC): An EVC is an association of two or more UNIs that limits the exchange of Service Frames to UNIs in the Ethernet Virtual Connection (EVC).

Operator Virtual Connection(OVC): An OVC is the association of UNIs and ENNIs or two ENNIs within one administrative domain.

This document uses the following abbreviations:

BSS: Business Support System

B-U-M: 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

VSI: Virtual Switching Instance

UNI: User to Network Interface

ENNI: External Network to Network Interface

3. The Layer 2 VPN Service Model

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

This document presents the L2SM using the YANG data modeling language [RFC6020] 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] and EVC service.

3.1. Applicability of the Layer 2 VPN Service Model

The L2SM defined in this document applies to VPW Service, VPLS service,EVPN service and Ethernet virtual circuit Services(e.g., E-Line and E-LAN service).

Over the past decade, The MEF Forum (MEF) has published a series of technical specifications of Ethernet virtual circuit service attributes and implementation agreements between providers. Many Ethernet VPN service providers worldwide have adopted these MEF standards and developed backoffice tools accordingly. IETF also works on extending L2VPN Framework [RFC4664] to support those services in the MPLS network.

Rather than introducing a new set of terminologies, the L2SM will align with existing MEF attributes when it's applicable to Ethernet Virtual Circuit Service. Therefore, service providers can easily integrate any new application that leverages the L2SM data using(for example, a Service Orchestrator), with existing BSS/OSS toolsets. Service providers also have the option to generate L2SM data for current L2VPN customer circuits already deployed in the network.

3.2. Layer 2 VPN Service Types

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

- Point-to-point Virtual Private Wire Services (VPWS);
- PWE3 (Pseudo-Wire Emulation Edge to Edge) Services that use LDP-signaled Pseudowires;
- Multipoint Virtual Private LAN services (VPLS) that use LDP-signaled Pseudowires;
- Multipoint Virtual Private LAN services (VPLS) that use a Border Gateway Protocol (BGP) control plane as described in RFC4761 and RFC6624;

- Ethernet VPNs specified in RFC 7432;

- EVC Service

An EVC circuit can also be port-based, in which case any service frames received from a subscriber within the contractual bandwidth will be delivered to the corresponding remote site, regardless of the customer VLAN value (C-tag) of the incoming frame. When multiple service frames are received from a subscriber and each service frame has different C-tag, all C-tags have to be mapped to one Ethernet Service(i.e., All to One bundling). The service frames can also be native Ethernet frames without a C-tag. In this scenario, only one Ethernet Virtual Circuit (EVC) is allowed on a single provider to subscriber link.

Contrary to the above use case, incoming customer service frames may be split into multiple EVCs based on pre-arrangement between the service provider and customer. Typically, C-tag of the incoming frames will serve as the service delimiter for EVC multiplexing over the same provider to subscriber interconnection.

Combining the port based attribute and service-multiplexing attribute with the connection type (point-to-point or multipoint-to-multipoint), an Ethernet Virtual Circuit may fall into one of the following service types:

- E-Line services: Point-to-Point Layer 2 connections.

  EPL: In its simplest form, a port-based Ethernet Private Line (EPL) service provides a high degree of transparency delivering all customer service frames between local and remote UNIs using multiple C-tags to one EVC bundling or All-to-One Bundling [MEF 6.1]. All unicast/broadcast/multicast packets are delivered unconditionally over the EVC. No service multiplexing is allowed on an EPL UNI. Note that The UNI interface connecting provider edge and customer edge devices is called an Attachment Circuit (AC) and can be a physical or virtual link.

  EVPL: On the other hand, a VLAN based Ethernet Virtual Private Line (EVPL) service supports multiplexing more than one point-to-point, or even other virtual private services, on the same UNI. Ingress service frames are conditionally transmitted through one of the EVCs based upon pre-agreed C-tag to EVC mapping. EVPL supports multiple C-tags to one EVC bundling.

- E-LAN services: Multipoint-to-Multipoint Layer 2 connections.

EP-LAN: An Ethernet Private LAN Service (EP-LAN) transparently connects multiple subscriber sites together with All-to-One Bundling. No service multiplexing is allowed on an EP-LAN UNI.

EVP-LAN: Some subscribers may desire more sophisticated control of data access between multiple sites. An Ethernet Virtual Private LAN Service (EVP-LAN) allows multiple EVCs to be connected to from one or more of the UNIs. Services frame disposition is based on C-tag to EVC mapping. EVP-LAN supports multiple C-tags bundled to one EVC.

3.3. 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. A L2VPN provides end-to-end L2 connectivity over this multi-service core infrastructure between two or more locations of Customers or a collection of sites. Attachment Circuit are placed between CE devices and PE Devices that backhaul service frames from the customer over the access network to the Provider Network or remote Site. The demarcation point (i.e.,UNI) between customer and service provider can be either placed between C and Customer Edge Device or between Customer Edge Device and Provider Edge Device. The actual bearer, connection, network access type between CE and PE will be discussed in the L2SM model.

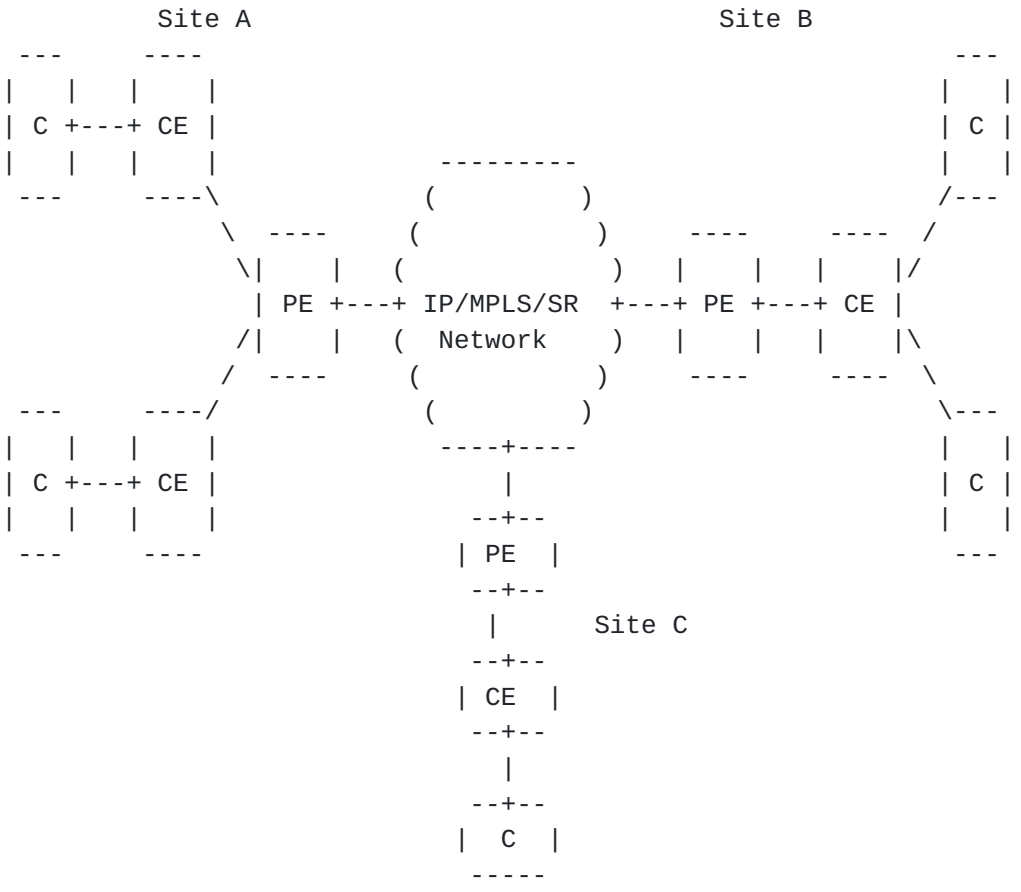


Figure 1: Reference Network for the Use of the L2VPN Service Model

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.

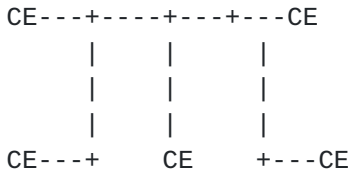


Figure 2: Customer View of the L2VPN

3.4. Layer 2 VPN Ethernet Virtual Circuit Construct

The base model of L2VPN EVC is shown in Figure 3.

Customer edge network device (CE) connects to the service provider's PE equipment. The demarcation point between customer and service

provider devices is referred as the User Network Interface (UNI). The UNI interface connecting PE and CE devices can be a physical or virtual port. For clarification, this is called the UNI-C on the customer side and UNI-N on the provider side.

The service provider is obligated to deliver the original service frame from local UNI-C across the network to the remote UNI-C. All Ethernet and IP header information, include (but not limit to) source and destination MAC addresses, EtherType, VLAN (C-tag), Class-of-Service marking (802.1p or DSCP), etc.

Incoming service frames are first examined at UNI-C based on C-tag, Class-of-Services identifier, EtherType value. Conforming packets are then metered against the contractual service bandwidth. Conforming packets will be delivered to the remote UNI via the Ethernet Virtual Circuit (EVC), which spans between UNI-C and UNI-C.

When both CEs are located in the same provider's network, a single operator maintains the EVC. In this case, the EVC consists of only one Operator Virtual Circuit (OVC).

Typically, the CE device at customer premises is a layer 2 Ethernet switch. A service provider may choose to impose an outer VLAN tag (S-tag) into the received subscriber traffic following 802.1ad Q-in-Q standard, especially when Layer 2 aggregation devices exist between CE and PE.

The uplink from PE to PE is referred as an Internal Network-to-Network Interface (I-NNI). When 802.1ad Q-in-Q is implemented, Ethernet frames from CE to PE are double tagged with both provider and subscriber VLANs (S-tag, C-tag).

Most service providers have deployed MPLS or SR multi-service core infrastructure. Ingress service frames will be mapped to either Ethernet Pseudowire (PWE) or VxLAN tunnel PE-to-PE. The details of these tunneling mechanism are at the provider's discretion and not part of the L2SM.

The service provider may also choose a Seamless MPLS approach to expand the PWE or VxLAN tunnel between UNI-N to UNI-N.

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

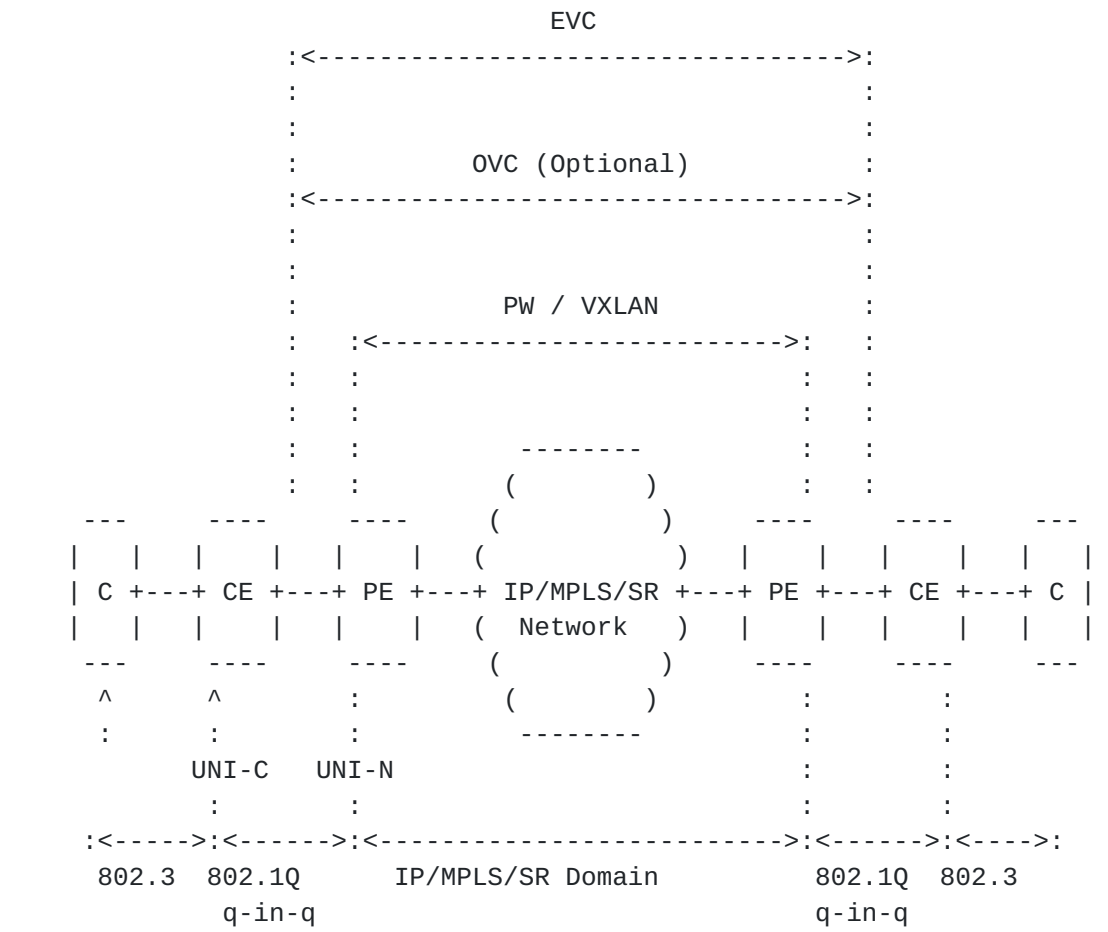


Figure 3: Architectural Model for EVC over a Single Network

Nevertheless, the remote site may be outside of the provider's service territory. In this case, the provider may partner with the operator of another metro network to provide service to the off-net location as shown in Figure 4.

The first provider owns the customer relationship, thus the end-to-end EVC. The EVC is comprised of two or more OVCs. The EVC is partially composed of an OVC from local UNI-C to the inter- provider interface. The provider will purchase an Ethernet Access (E-Access) OVC from the second operator to deliver packet to the remote UNI-C.

The inter-connect between the two operators edge gateway (EG) devices is defined as the External Network-to-Network Interface (E-NNI).

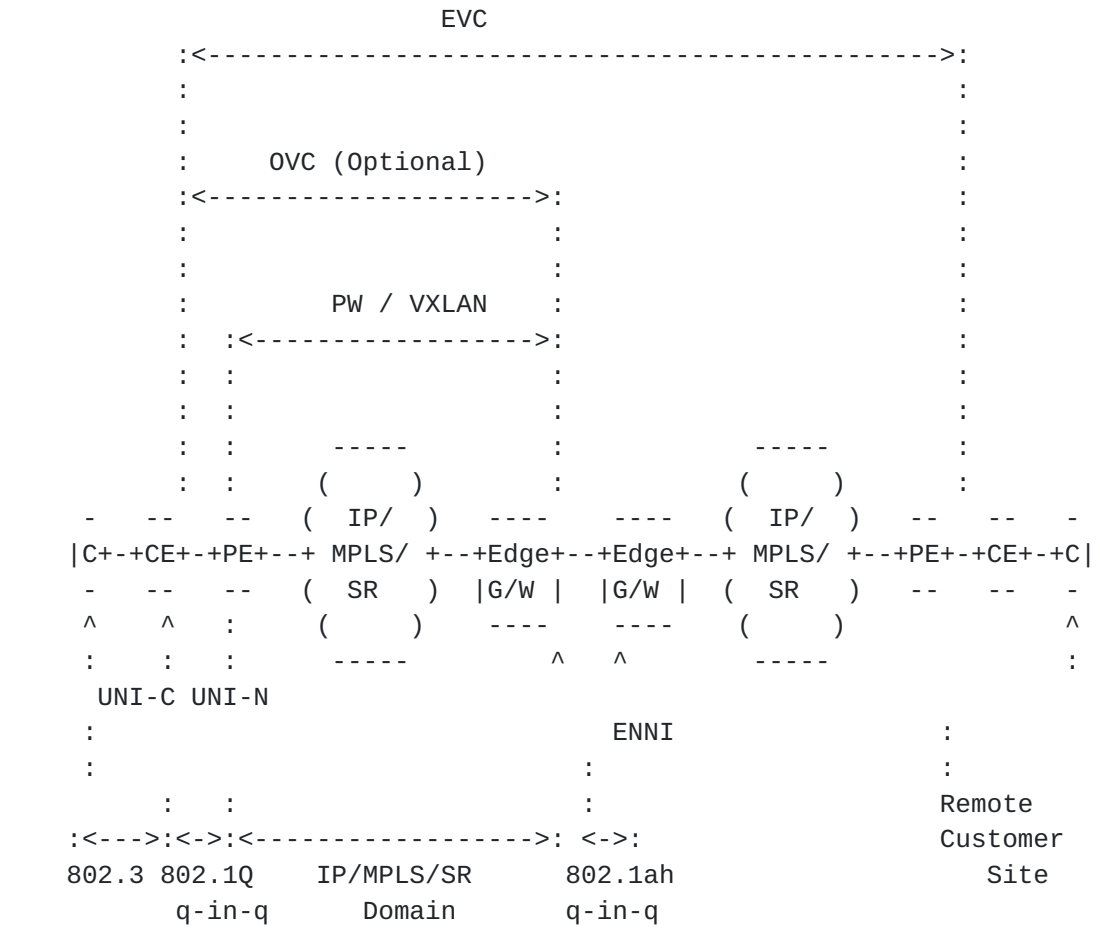


Figure 4: Architectural Model for EVC over Multiple Networks

4. Service Data Model Usage

The L2VPN service model provides an abstracted interface to request, configure, and manage the components of a 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 to be 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 5 and described in more detail in [I-D.ietf-opsawg-service-model-explained]. 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 [RFC8049].

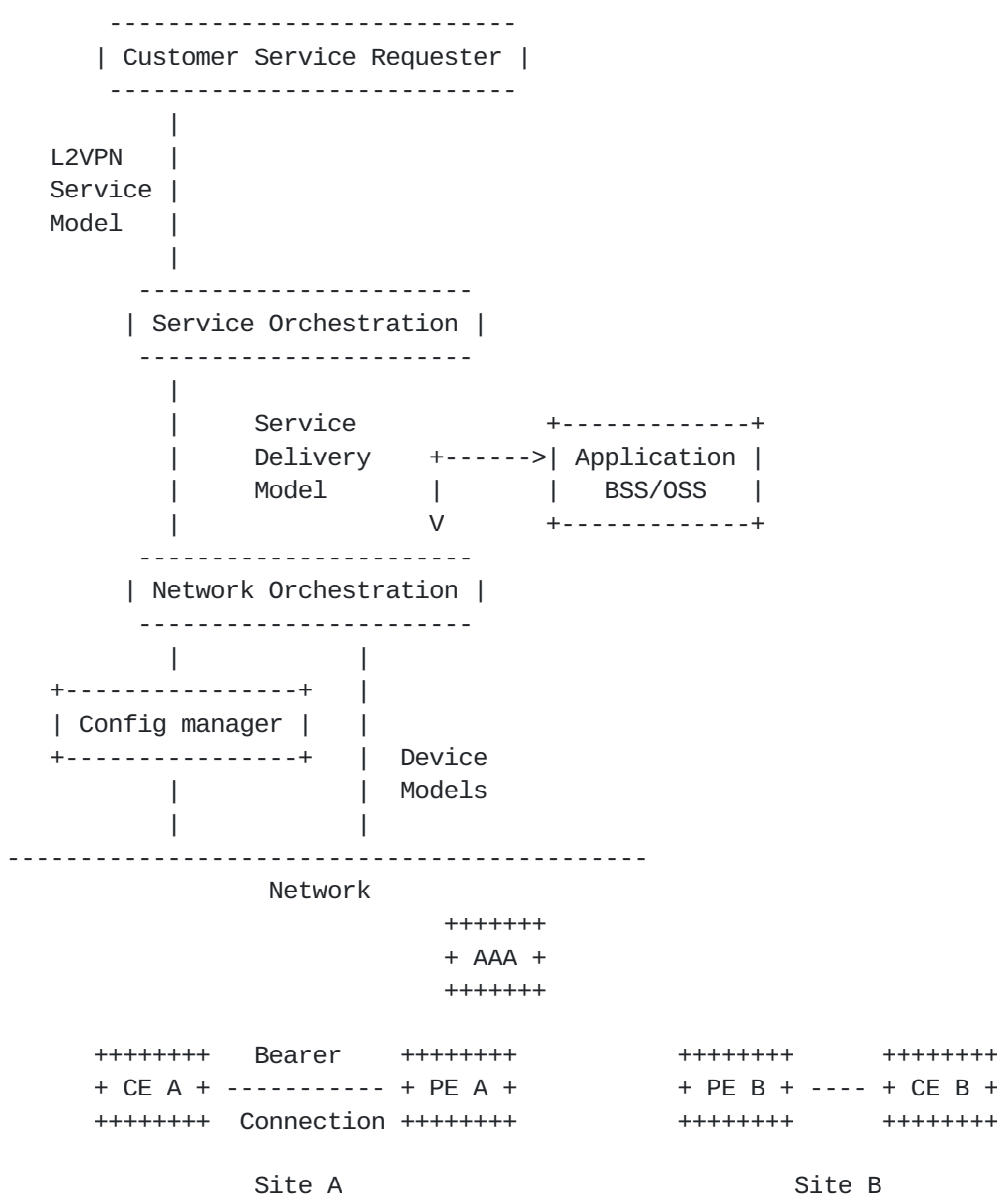


Figure 5: Reference Architecture for the Use of the L2VPN Service Model

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.

The YANG module is divided in 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 5.3.4) 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 combination of Per Site connectivity and OVC at different segments.

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

```
module: ietf-l2vpn-svc
   +--rw l2vpn-svc
      +--rw vpn-services
      |  +--rw vpn-svc* [vpn-id]
      |     +--rw vpn-id                    svc-id
      |     +--rw vpn-type?                 identityref
      |     +--rw customer-account-number?  uint32
      |     +--rw customer-name?            string
      |     +--rw evc {evc}?
      |     |  +--rw enabled?          boolean
      |     |  +--rw evc-type?         identityref
      |     |  +--ro number-of-pe?     uint32
      |     |  +--ro number-of-site?   uint32
      |     |  +--rw uni-list {uni-list}?
      |     |     +--rw uni-list* [uni-site-id]
      |     |        +--rw uni-site-id                string
      |     |        +--rw off-net?                   boolean
      |     |        +--rw service-multiplexing?      boolean
      |     |        +--rw ce-vlan-preservation?      boolean
      |     |        +--rw ce-vlan-cos-perservation?  boolean
      |     +--rw ovc {ovc}?
      |     |  +--rw ovc-list* [ovc-id]
      |     |     +--rw ovc-id                   svc-id
      |     |     +--rw off-net?                 boolean
      |     |     +--rw svlan-cos-preservation?  boolean
      |     |     +--rw svlan-id-preservation?   boolean
```

```
      |     |     +--rw svlan-id-ethernet-tag?    string
      |     |     +--rw ovc-endpoint?             string
      |     +--rw svc-topo?                 identityref
      |     +--rw cloud-accesses {cloud-access}?
      |     |  +--rw cloud-access* [cloud-identifier]
      |     |     +--rw cloud-identifier    string
      |     |     +--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 authorized-sites
      |     |     |  +--rw authorized-site* [site-id]
      |     |     |     +--rw site-id    -> /l2vpn-svc/sites/site/site-id
      |     |     +--rw denied-sites
      |     |        +--rw denied-site* [site-id]
      |     |           +--rw site-id    -> /l2vpn-svc/sites/site/site-id
      |     +--rw metro-networks
      |     |  +--rw metro-network* [id]
      |     |     +--rw id                    string
      |     |     +--rw inter-mkt-service?   boolean
      |     |     +--rw intra-mkt* [mkt-name]
      |     |        +--rw mkt-name        string
      |     |        +--rw ovc-id?         -> /l2vpn-svc/vpn-services/vpn-svc/
ovc/ovc-list/ovc-id
      |     |        +--rw site-id?        -> /l2vpn-svc/sites/site/site-id
      |     +--rw global-l2cp-control {L2CP-control}?
      |     |  +--rw stp-rstp-mstp?   control-mode
      |     |  +--rw pause?           control-mode
      |     |  +--rw lldp?            boolean
      |     +--rw service-level-mac-limit
      |     |  +--rw service-level-mac-limit?   uint16
      |     |  +--rw action?                    identityref
      |     +--rw service-protection
      |     |  +--rw protection-mode?   identityref
      |     +--rw cvlan-id-to-svc-map* [svc-id type]
      |     |  +--rw svc-id      -> /l2vpn-svc/vpn-services/vpn-svc/vpn-id
      |     |  +--rw type        identityref
      |     |  +--rw cvlan-id* [vid]
      |     |     +--rw vid    identityref
      |     +--rw multicast {multicast}?
      |     |  +--rw enabled?                boolean
      |     |  +--rw customer-tree-flavors
      |     |  |  +--rw tree-flavor*   identityref
```

```
|     |  +--rw traffic-type?            identityref
|     |  +--rw group-port-mapping?      identityref
|     +--rw extranet-vpns {extranet-vpn}?
```

```
      |        +--rw extranet-vpn* [vpn-id]
      |           +--rw vpn-id             svc-id
      |           +--rw local-sites-role?   identityref
      +--rw sites
         +--rw site* [site-id site-type]
            +--rw site-id                                string
            +--rw site-type                              identityref
            +--rw device
            |  +--rw devices* [device-id]
            |     +--rw device-id     string
            |     +--rw location?     -> /l2vpn-svc/sites/site/locations/
location/location-id
            |     +--rw management
            |        +--rw address?                inet:ip-address
            |        +--rw management-transport?   identityref
            +--rw locations
            |  +--rw location* [location-id]
            |     +--rw location-id     string
            |     +--rw address?        string
            |     +--rw zip-code?       string
            |     +--rw state?          string
            |     +--rw city?           string
            |     +--rw country-code?   string
            +--rw management
            |  +--rw type?   identityref
            +--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 ipv4-lan-prefix*   inet:ipv4-prefix {ipv4}?
            |        |     +--rw ipv6-lan-prefix*   inet:ipv6-prefix {ipv6}?
            |        +--rw vpn* [vpn-id]
            |           +--rw vpn-id       -> /l2vpn-svc/vpn-services/vpn-svc/
vpn-id
            |           +--rw site-role?   identityref
            +--rw signaling-options {signaling-options}?
            |  +--rw signaling-options* [type]
            |     +--rw type           identityref
            |     +--rw bgp-l2vpn
```

```
            |     |  +--rw vpn-id?                    -> /l2vpn-svc/vpn-
services/vpn-svc/vpn-id
            |     |  +--rw type?                      identityref
```

```
            |     |  +--rw pwe-encapsulation-type?   identityref
            |     |  +--rw pwe-mtu
            |     |  |  +--rw allow-mtu-mismatch?   boolean
            |     |  +--rw address-family?           identityref
            |     +--rw mp-bgp-evpn
            |     |  +--rw vpn-id?              -> /l2vpn-svc/vpn-services/vpn-
svc/vpn-id
            |     |  +--rw type?                identityref
            |     |  +--rw address-family?      identityref
            |     |  +--rw mac-learning-mode?   identityref
            |     |  +--rw arp-suppress?        boolean
            |     +--rw t-ldp-pwe
            |     |  +--rw type?                      identityref
            |     |  +--rw pwe-encapsulation-type?   identityref
            |     |  +--rw pwe-mtu
            |     |  |  +--rw allow-mtu-mismatch?   boolean
            |     |  +--rw pe-eg-list* [service-ip-addr vc-id]
            |     |  |  +--rw service-ip-addr    inet:ip-address
            |     |  |  +--rw vc-id              string
            |     |  |  +--rw peer-id?           string
            |     |  |  +--rw remote-peer-id?    string
            |     |  |  +--rw pw-priority?       uint32
            |     |  +--rw control-word?             boolean
            |     |  +--rw qinq
            |     |     +--rw s-tag?   uint32
            |     |     +--rw c-tag?   uint32
            |     +--rw l2tp-pw
            |        +--rw encapsulation-type?   identityref
            |        +--rw control-word?         boolean
            +--rw load-balance-options
            |  +--rw enable?               boolean
            |  +--rw load-balance-method?   identityref
            +--rw service
            |  +--rw qos {qos}?
            |     +--rw qos-classification-policy
            |     |  +--rw rule* [id]
            |     |     +--rw id                uint16
            |     |     +--rw (match-type)?
            |     |     |  +--:(match-flow)
            |     |     |  |  +--rw match-flow
            |     |     |  |     +--rw dscp?           inet:dscp
            |     |     |  |     +--rw dot1p?          uint8
            |     |     |  |     +--rw pcp?            uint8
            |     |     |  |     +--rw src-mac?        yang:mac-address
            |     |     |  |     +--rw dst-mac?        yang:mac-address
            |     |     |  |     +--rw composite-id
            |     |     |  |     |  +--rw endpoint-id?   string
            |     |     |  |     |  +--rw cos-label?     identityref
```

```
|     |     |  |     |  +--rw pcp?           uint8
```

```
|     |     |  |     |  +--rw dscp?          inet:dscp
|     |     |  |     +--rw color-type?     identityref
|     |     |  |     +--rw target-sites*   svc-id
|     |     |  +--:(match-phy-port)
|     |     |     +--rw match-phy-port?    uint16
|     |     +--rw target-class-id?   string
|     +--rw qos-profile
|        +--rw (qos-profile)?
|           +--:(standard)
|           |  +--rw profile?   string
|           +--:(custom)
|              +--rw classes {qos-custom}?
|                 +--rw class* [class-id]
|                    +--rw class-id              string
|                    +--rw direction?            identityref
|                    +--rw policing?             identityref
|                    +--rw byte-offset?          uint16
|                    +--rw rate-limit?           uint8
|                    +--rw discard-percentage?   uint8
|                    +--rw frame-delay
|                    |  +--rw (flavor)?
|                    |     +--:(lowest)
|                    |     |  +--rw use-low-del?   empty
|                    |     +--:(boundary)
|                    |        +--rw delay-bound?   uint16
|                    +--rw frame-jitter
|                    |  +--rw (flavor)?
|                    |     +--:(lowest)
|                    |     |  +--rw use-low-jit?   empty
|                    |     +--:(boundary)
|                    |        +--rw delay-bound?   uint32
|                    +--rw frame-loss
|                    |  +--rw fr-loss-rate?   decimal64
|                    +--rw bandwidth
|                       +--rw guaranteed-bw-percent?   uint8
|                       +--rw end-to-end?              empty
+--rw broadcast-unknown-unicast-multicast
|  +--rw multicast-site-type?           enumeration
|  +--rw multicast-gp-address-mapping* [id]
|  |  +--rw id                uint16
|  |  +--rw vlan-id?          uint32
|  |  +--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 security
```

```
|  +--rw mac-loop-prevention
|  |  +--rw frequency?         uint32
|  |  +--rw protection-type?   identityref
|  |  +--rw number-retries?    uint32
|  +--rw access-control-list
|  |  +--rw mac* [mac-address]
|  |     +--rw mac-address    yang:mac-address
|  +--rw mac-addr-limit
|     +--rw exceeding-option?   uint32
+--ro actual-site-start?                    yang:date-and-time
+--ro actual-site-stop?                     yang:date-and-time
+--rw site-network-accesses
   +--rw site-network-accesse* [network-access-id]
      +--rw network-access-id                     string
      +--rw remote-carrier-name?                  string
      +--rw access-diversity {site-diversity}?
      |  +--rw groups
      |  |  +--rw fate-sharing-group-size?   uint16
      |  |  +--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 requested-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 dot1q-interface
      |  |  +--rw l2-access-type?   identityref
      |  |  +--rw dot1q {dot1q}?
      |  |  |  +--rw physical-inf?   string
      |  |  |  +--rw c-vlan-id?      uint32
      |  |  +--rw qinq {qinq}?
      |  |  |  +--rw s-vlan-id?   uint32
```

```
|  |  |  +--rw c-vlan-id?   uint32
|  |  +--rw qinany {qinany}?
|  |  |  +--rw s-vlan-id?   uint32
|  |  +--rw vxlan {vxlan}?
|  |     +--rw vni-id?      uint32
|  |     +--rw peer-mode?   identityref
|  |     +--rw peer-list* [peer-ip]
|  |        +--rw peer-ip    inet:ip-address
|  +--rw phy-interface
|  |  +--rw port-number?           uint32
|  |  +--rw port-speed?            uint32
|  |  +--rw mode?                  neg-mode
|  |  +--rw phy-mtu?               uint32
|  |  +--rw flow-control?          string
|  |  +--rw physical-if?           string
|  |  +--rw circuit-id?            string
|  |  +--rw lldp?                  boolean
|  |  +--rw oam-802.3ah-link {oam-3ah}?
|  |  |  +--rw enable?   boolean
|  |  +--rw uni-loop-prevention?   boolean
|  +--rw lag-interface {lag-interface}?
|  |  +--rw lag-interface* [lag-interface-number]
|  |     +--rw lag-interface-number    uint32
|  |     +--rw lacp
|  |        +--rw lacp-state?         boolean
|  |        +--rw lacp-mode?          boolean
|  |        +--rw lacp-speed?         boolean
|  |        +--rw mini-link?          uint32
|  |        +--rw system-priority?    uint16
|  |        +--rw micro-bfd {micro-bfd}?
|  |        |  +--rw micro-bfd-on-off?   enumeration
|  |        |  +--rw bfd-interval?       uint32
|  |        |  +--rw bfd-hold-timer?     uint32
|  |        +--rw bfd {bfd}?
|  |        |  +--rw bfd-enabled?    boolean
|  |        |  +--rw (holdtime)?
|  |        |     +--:(profile)
|  |        |     |  +--rw profile-name?   string
|  |        |     +--:(fixed)
|  |        |        +--rw fixed-value?    uint32
|  |        +--rw member-link-list
|  |        |  +--rw member-link* [name]
|  |        |     +--rw name                string
|  |        |     +--rw port-speed?         uint32
|  |        |     +--rw mode?               neg-mode
|  |        |     +--rw mtu?                uint32
|  |        |     +--rw oam-802.3ah-link {oam-3ah}?
|  |        |        +--rw enable?   boolean
```

```
                  |  |        +--rw flow-control?       string
                  |  |        +--rw lldp?               boolean
                  |  +--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 provider-bridge-group?  control-mode
                  |     +--rw provider-bridge-mvrp?   control-mode
                  +--rw svc-mtu?                              uint32
                  +--rw availability
                  |  +--rw access-priority?   uint32
                  |  +--rw (redundancy-mode)?
                  |     +--:(single-active)
                  |     |  +--rw single-active?     boolean
                  |     +--:(all-active)
                  |        +--rw all-active?        boolean
                  +--rw vpn-attachment
                  |  +--rw device-id?       string
                  |  +--rw management
                  |  |  +--rw address-family?   identityref
                  |  |  +--rw address?          inet:ip-address
                  |  +--rw (attachment-flavor)
                  |     +--:(vpn-flavor)
                  |     |  +--rw vpn-flavor* [vpn-id]
                  |     |     +--rw vpn-id       -> /l2vpn-svc/vpn-services/
vpn-svc/vpn-id
                  |     |     +--rw site-role?   identityref
                  |     +--:(vpn-policy-id)
                  |        +--rw vpn-policy-id?   -> /l2vpn-svc/sites/site/vpn-
policies/vpn-policy/vpn-policy-id
                  +--rw service
                  |  +--rw svc-input-bandwidth {input-bw}?
                  |  |  +--rw input-bandwidth* [type]
                  |  |     +--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?      uint32
```

```
|  |     +--rw pbs?      uint32
|  +--rw svc-output-bandwidth {output-bw}?
```

```
|  |  +--rw output-bandwidth* [type]
|  |     +--rw type      identityref
|  |     +--rw cos-id?   uint8
|  |     +--rw vpn-id?   svc-id
|  |     +--rw cir?      uint32
|  |     +--rw cbs?      uint32
|  |     +--rw eir?      uint32
|  |     +--rw ebs?      uint32
|  |     +--rw pir?      uint32
|  |     +--rw pbs?      uint32
|  +--rw qos {qos}?
|     +--rw qos-classification-policy
|     |  +--rw rule* [id]
|     |     +--rw id                 uint16
|     |     +--rw (match-type)?
|     |     |  +--:(match-flow)
|     |     |  |  +--rw match-flow
|     |     |  |     +--rw dscp?           inet:dscp
|     |     |  |     +--rw dot1p?          uint8
|     |     |  |     +--rw pcp?            uint8
|     |     |  |     +--rw src-mac?        yang:mac-address
|     |     |  |     +--rw dst-mac?        yang:mac-address
|     |     |  |     +--rw composite-id
|     |     |  |     |  +--rw endpoint-id?   string
|     |     |  |     |  +--rw cos-label?     identityref
|     |     |  |     |  +--rw pcp?           uint8
|     |     |  |     |  +--rw dscp?          inet:dscp
|     |     |  |     +--rw color-type?     identityref
|     |     |  |     +--rw target-sites*   svc-id
|     |     |  +--:(match-phy-port)
|     |     |     +--rw match-phy-port?    uint16
|     |     +--rw target-class-id?   string
|     +--rw qos-profile
|        +--rw (qos-profile)?
|           +--:(standard)
|           |  +--rw profile?   string
|           +--:(custom)
|              +--rw classes {qos-custom}?
|                 +--rw class* [class-id]
|                    +--rw class-id              string
|                    +--rw direction?            identityref
|                    +--rw policing?             identityref
|                    +--rw byte-offset?          uint16
|                    +--rw rate-limit?           uint8
|                    +--rw discard-percentage?   uint8
|                    +--rw frame-delay
|                    |  +--rw (flavor)?
|                    |     +--:(lowest)
```

```
|                    |     |  +--rw use-low-del?   empty
|                    |     +--:(boundary)
|                    |        +--rw delay-bound?   uint16
|                    +--rw frame-jitter
|                    |  +--rw (flavor)?
|                    |     +--:(lowest)
|                    |     |  +--rw use-low-jit?   empty
|                    |     +--:(boundary)
|                    |        +--rw delay-bound?   uint32
|                    +--rw frame-loss
|                    |  +--rw fr-loss-rate?   decimal64
|                    +--rw bandwidth
|                       +--rw guaranteed-bw-percent?   uint8
|                       +--rw end-to-end?              empty
|                 +--rw broadcast-unknown-unicast-multicast
|                 |  +--rw multicast-site-type?           enumeration
|                 |  +--rw multicast-gp-address-mapping* [id]
|                 |  |  +--rw id                 uint16
|                 |  |  +--rw vlan-id?           uint32
|                 |  |  +--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 ethernet-service-oam
|                 |  +--rw md-name?        string
|                 |  +--rw md-level?       uint8
|                 |  +--rw cfm-802.1-ag
|                 |  |  +--rw n2-uni-c* [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 n2-uni-n* [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 security
   +--rw mac-loop-prevention
   |  +--rw frequency?         uint32
   |  +--rw protection-type?   identityref
   |  +--rw number-retries?    uint32
   +--rw access-control-list
   |  +--rw mac* [mac-address]
   |     +--rw mac-address    yang:mac-address
   +--rw mac-addr-limit
      +--rw exceeding-option?   uint32
```

Figure 6

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 5.3.3.2) proposed to the customer may also be enabled through features. This model also proposes some features for options that are more advanced, such as support for extranet VPNs (Section 5.2.6), site diversity (Section 5.6), 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 proposes VXLAN [RFC7348] for Ethernet packet Encapsulation; if VXLAN Encapsulation do not fulfill all requirements, new options can be added through augmentation.

5.2. VPN Service Overview

A vpn-service list item contains generic informations 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 Type (vpn-type): Used to indicate VPN service Type. The identifier is a string allowing to any encoding for the local administration of the VPN service.

Ethernet Connection Service Type (evc-type): used to identify supported Ethernet Connection Service Types in case of EVC service being offered to the customer.

Cloud Access (cloud-access): All sites in the L2VPN MUST be authorized to access to the cloud.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 is required for configuration.

Metro Network Partition: Used by service provide to divide the network into several administrative domains.

Load Balance (load-balance-option): Intended to capture the load-balance agreement between the subscriber and provider.

CVLAN ID To EVC MAP: Contains the list of customer vlans to the service mapping in a free-form format. In most cases, this will be the VLAN access-list for the inner 802.1q tags.

Service Level MAC Limit: Contains the subscriber MAC address limit and exceeding action information.

Service Protection (svc-protection): Capture the desired service protection agreement between subscriber and provider.

Multicast Service (multicast): rovide multicast support for L2VPN.

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

5.2.1. Customer Information

The customer information contains two essential information to identify the subscriber.

"customer-account-number" is an internal alphanumerical number assigned by the service provider to identify the subscriber. It MUST be unique within the service provider's OSS/BSS system. The actual format depends on the system tool the provider uses. "customer-name" is in a more readable form and refers to a more-explicit reference to the customer. Both identifiers are purely internal to the organization responsible for the VPN service.

5.2.2. VPN Service Type

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

- Point-to-point Virtual Private Wire Services (VPWS);
- PWE3 (Pseudo-Wire Emulation Edge to Edge) that use LDP-signaled Pseudowires;
- Multipoint Virtual Private LAN services (VPLS) that use LDP-signaled Pseudowires;
- Multipoint Virtual Private LAN services (VPLS) that use a Border Gateway Protocol (BGP) control plane as described in RFC4761 and RFC6624;
- Ethernet VPNs specified in RFC 7432;
- Ethernet Private Line (EPL) Service with PW core;
- Ethernet Virtual Private Line (EVPL) Service with PW core;
- Ethernet Private LAN (EP-LAN)Service with VPLS core;
- Ethernet Virtual Private LAN (EVP-LAN)Service with VPLS core

Other L2VPN Service Type could be included by augmentation. Note that EPL service and EVPL service are E-Line service or point to point EVC service while EP-LAN service and EVP-LAN service are E-LAN service or multiple point to multipoint EVC service.

5.2.2.1. EVC

The "evc" container contains "enable" leafand "uni-list" container. If EVC service for Provider provision L2VPN is required, the "enabled" leaf MUST be set to true in the "evc" container. "uni-list" will specify the UNI list associated with this EVC service.

In addition, "evc-type","number of PEs" and "number of sites" can be specified under the "evc" container.The "evc-type" defines three EVC service types: Point-to-Point,Multipoint-to-Multipoint, Rooted-Multipoint. New Ethernet Connection service types can be added by augmentation in the future. "number of PEs" and "number of sites" describes the number of PEs and number of sites along with EVC connection.

E-Line and E-LAN providers shall have an EVC-ID assigned to the UNI-to-UNI circuit.EVC-ID value will be set to the same VPN-id value under vpn-service list.

5.2.2.2. OVC

The "ovc-list" under "ovc" container defines a list of "ovc-id" parameter associated with "evc". The "off-net" leaf MUST be set to true if one of external interface of "ovc" is UNI and this UNI can not be reachable by the customer or local site.

For E-Access service as an OVC-based service, the "off-net" leaf MUST be marked false(i.e., on-net is enabled), and The E-Access service provider will assign an "ovc-id" for the circuit between UNI and E-NNI.

If the service is E-Line or E-LAN with remote UNIs, there will be one, and only one, on-net "ovc-id" and a list of off-net "ovc-id" objects for the remote UNIs.

Service providers have the option of inserting an outer VLAN tag (the S-tag) into the service frames from the customer to improve service scalability and customer VLAN transparency.

The "svlan-id-ethernet-tag" is either the S-tag inserted at a UNI or the outer tag of ingress packets at an E-NNI. This parameter is included in the L2SM model to facilitate other management system to generate proper configuration for the network elements.

Ideally, all external interfaces (UNI and E-NNI) associated with a given service will have the same S-tag assigned. However, this may not always be the case. Traffic with all attachments using different S-tags will need to be "normalized" to a single service S-tag. (One

example of this is a multipoint service that involves multiple off-net OVCs terminating on the same E-NNI. Each of these off-net OVCs will have a distinct S-tag which can be different from the S-tag used in the on-net part of the service.)

S-VLAN ID Preservation and S-VLAN CoS Preservation apply between two ENNIs connected by an OVC. These two attributes do NOT affect ENNI to UNI frame exchange. Preservation means that the value of S-VLAN ID and/or S-VLAN CoS at one ENNI must be equal to the value at a different ENNI connected by the OVC. The purpose of the optional "svlan-id-ethernet-tag" leaf is to identify the service-wide "normalized S-tag".If optional "svlan-id-perservation" leaf is set to true, the "svlan-id-ethernet-tag" leaf MUST be configured.

5.2.3. 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). New topologies could be added by augmentation. By default, the any-to-any VPN service topology is used.

5.2.3.1. Route Target Allocation

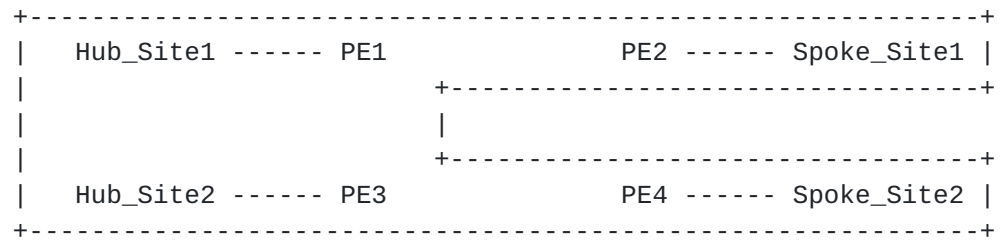
A Layer 2 PE-based VPN (such as VPLS based VPN or EVPN that uses BGP as signaling protocol) can be built using route targets (RTs) as described in [RFC4364][RFC7432]. The management system is expected to automatically allocate a set of RTs upon receiving a VPN service creation request. How the management 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 [RFC8049].

5.2.3.2. 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 VFI/VSI/EVI and RTs on the appropriate PEs. In the any-to-any case, a single RT is generally required, and every VFI/VSI/EVI imports and exports this RT.

5.2.3.3. 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 owns an any-to-any L2 VPN service request through this model is expected to assign and then configure the VFI/VSI/EVI 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 VFI/VSI/EVI 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 VFI/VSI/EVI that connects Spoke sites will export Spoke routes with the Spoke RT and will import Hub routes through the Hub RT.

5.2.4. Cloud Access

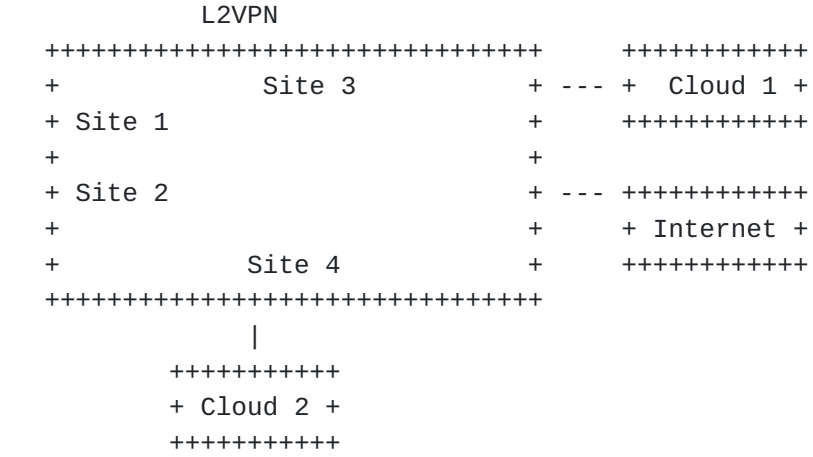
This model provides cloud access configuration through the cloud-access container. The usage of cloud-access is targeted for public cloud and Internet Access. The cloud-access container provides parameters for authorization rules.

Private cloud access may be addressed through the site contianer as described in Section 5.3 with the use consistent with sites of type E-NNI.

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

By default, all sites in the IP VPN MUST be authorized to access the cloud. If restrictions are required, a user MAY configure 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 pointing to the cloud identifier for the Internet service. No authorized sites will be configured, as all sites are required to access the Internet.

```
<vpn-service>
    <vpn-id>123456487</vpn-id>
    <cloud-accesses>
     <cloud-access>
        <cloud-identifier>INTERNET</cloud-identifier>
     </cloud-access>
    </cloud-accesses>
</vpn-service>
```

If Site 1 and Site 2 require access to Cloud 1, a new cloud-access 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.

```
<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>
</vpn-service>
```

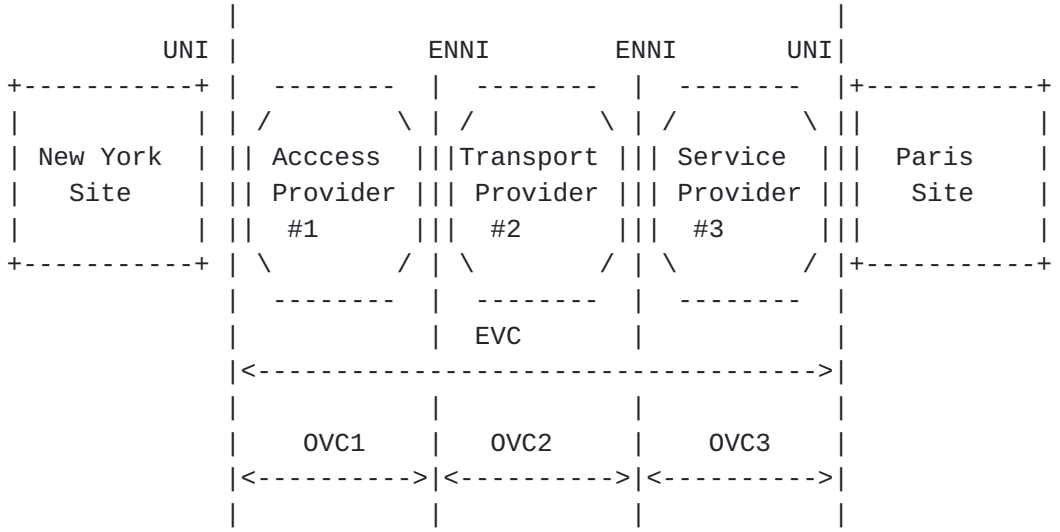
If all sites except Site 1 require access to Cloud 2, a new cloud-access pointing to the cloud identifier of Cloud 2 will be created. The deny-site leaf-list will be filled with a reference to Site 1.

```
<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>
</vpn-service>
```

5.2.5. Metro Ethernet Network Partition

Some service providers may divide their Metro Ethernet network into multiple administrative domains. And a EVC service may span across multiple such administrative domains belonging to the same service provider and be concatenated by one or multiple OVC segments. Each administrative domain has corresponding OVC segment. The optional "metro-networks" container is intended be used by these multi-domain providers to differentiate intra-market versus inter-market services.

When the "inter-mkt-service" leaf is marked TRUE, multiple associated "metro-mkt-id"s will be listed. Otherwise, the service is intra-domain and only one "metro-mkt-id" is allowed. In addition, "ovc-id""site-id" can be used to describe OVC and Site associated with the specific "metro-mkt-id".



In the example below, New York Site want to connect Paris Site across 3 adminstrative domains, with OVC in each domain:

```
   <vpn-service>
        <vpn-id>12456487</vpn-id>
        <evc>
            <uni-list>
                 <uni>UNI-Paris</uni>
                 <uni>UNI-NewYork</uni>
           </uni-list>
        </evc>
        <ovc>
           <ovc-list>
               <ovc>ovc1</ovc>
               <ovc>ovc2</ovc>
               <ovc>ovc3</ovc>
         </ovc-list>
        </ovc>
        <metro-networks>
              <mero-network>
                 <id>1</id>
                 <inter-mkt-service>TRUE</inter-mkt-service>
                 <intra-mkt>
                    <mkt-name>Access-Provider#1</mkt-name>
                    <ovc>ovc1</ovc>
                 </intra-mkt>
                 <intra-mkt>
                    <mkt-name>Transport-Provider#2</mkt-name>
                    <ovc>ovc2<ovc>
                 </intra-mkt>
                 <intra-mkt>
                    <mkt-name>Service-Provider#1</mkt-name>
                    <ovc>ovc3<ovc>
               </intra-mkt>
                    </metro-network>
         </metro-networks>
   </vpn-service>
```

5.2.6. 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.

```
                  +-----------+           +-----------+
                 /             \         /             \
     Site A -- |    VPN A      |  ---  |    VPN B      | --- Site B
                \             /         \             / (Shared
                  +-----------+           +-----------+   resources)
```

In the figure above, VPN B has some resources on Site B that need to be available to some customers/partners. 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 must be 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.

```
<vpn-service>
    <vpn-id>VPNB</vpn-id>
    <vpn-service-topology>hub-spoke</vpn-service-topology>
</vpn-service>
<vpn-service>
    <vpn-id>VPNA</vpn-id>
    <vpn-service-topology>any-to-any</vpn-service-topology>
    <extranet-vpns>
        <extranet-vpn>
            <vpn-id>VPNB</vpn-id>
            <site-role>spoke-role</site-role>
        </extranet-vpn>
    </extranet-vpns>
</vpn-service>
```

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

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.

5.2.7. 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 VPN 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 set of CLAN is bundled into one VPN service.

"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).

An EVC can be set to preserve the CE-VLAN ID and CE-VLAN CoS from ingress to egress. This is required when the customer is using the VLAN header information between its locations. CE-VLAN ID Preservation and CE-VLAN CoS Preservation apply between two UNIs connected by EVC. Preservation means that the value of CE-VLAN ID and/or CE-VLAN CoS at one UNI must be equal to the value at a different UNI connected by the same EVC.

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 CE-VLAN ID.

5.2.8. Service Level MAC Limit

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, and may 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.

The optional "service-level-mac-limit" container contains the customer MAC address limit and information to describe the action when the limit is exceeded.

5.2.9. Service Protection

The optional "service-protection" container is used to capture the desired service protection agreement between customer and provider.

Sometimes the customer may desire end-to-end protection at the service level for applications with high availability requirements. There are two protection schemes to offer redundant services:

- 1+1 protection: In this scheme, the primary circuit will be protected by a backup circuit, typically meeting certain diverse path/fiber/site/node criteria. Both primary and protection circuits are provisioned to be in the active forwarding state. The customer may choose to send the same service frames across both circuits simultaneously.

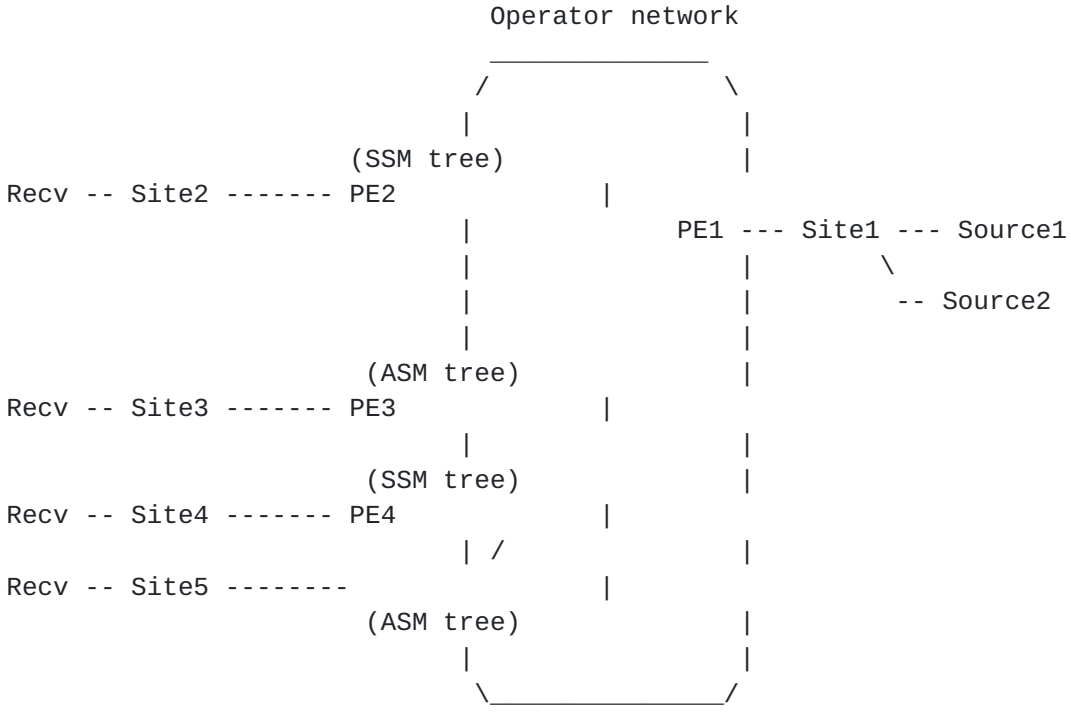
- 1:1 protection: In this scheme, a backup circuit to the primary circuit is provisioned. Depending on the implementation agreement, the protection circuits may either always be in active forwarding state, or may only become active when a faulty state is detected on the primary circuit.

5.2.10. Multicast Service

Multicast in L2VPNs is described in [RFC5501][RFC7117].

If multicast support is required for an L2VPN, some global multicast parameters are required as input for the service request. When a CE sends (1) Broadcast, (2) Multicast, or (3) Unknown destination unicast, replication occurs at ingress PE, therefore three traffic type is supported.

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



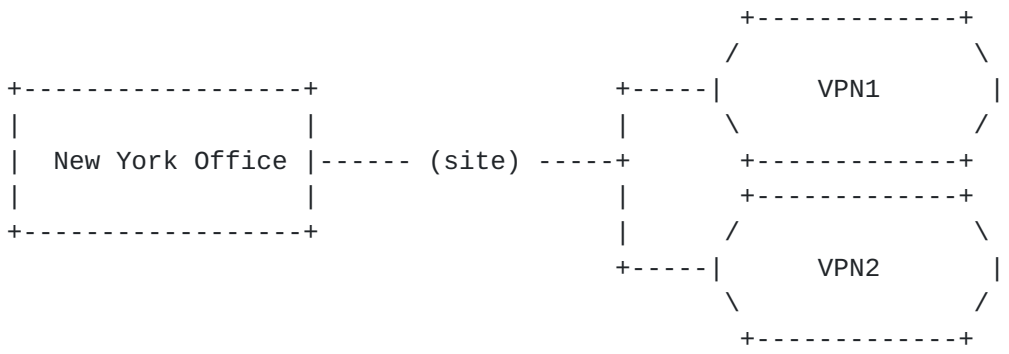
Group to port mappings can be created using the "rp-group-mappings" leaf. Two group to port mapping method are supported:

- 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) [802.1D].

5.3. Site Overview

A site represents a connection of a customer office to one or more VPN services.



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

We are restricting the L2SM to exterior interfaces 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.

Site Type (site-type): Defines the way the VPN multiplexing is done.

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 of the site, for example: type, management-transport, address.

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

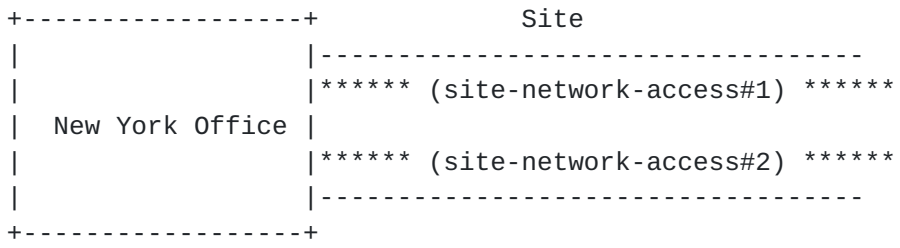
Site diversity (site-diversity): Presents some parameters to support site diversity.

Signaling Options(signaling-options): Defines which protocol or signaling must be activated between the customer and the provider.

Load balancing (load-balance-options): Defines the load-balancing agreement information between the customer and provider.

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

A site-network-access represents an Ethernet logical connection of 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.

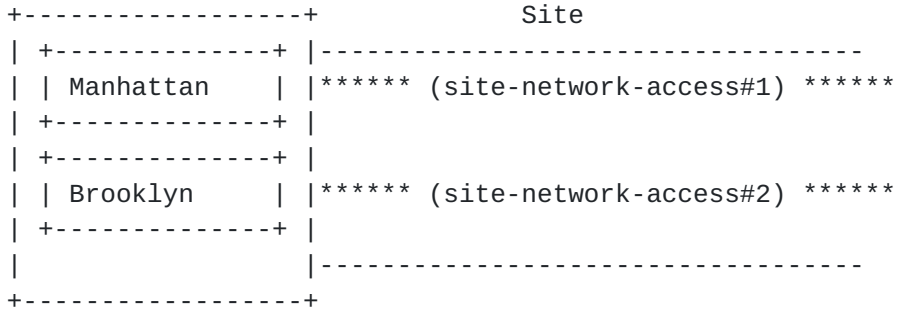
The site configuration is viewed as a global entity; we assume that it is mostly the management system's role to split the parameters between the different elements within the network. For example, in the case of the site-network-access configuration, the management system needs to split the overall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

5.3.1. Devices and Locations

The information in the "location" sub-container under a "site"and "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 component to choose the targeted upstream PE and downstream CE. Location is expressed in terms of postal information.

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 (customer metropolitan area network). In such a case, when connecting to a VPN service, the customer may ask for multihoming based on its distributed locations.

New York Site



A customer may also request 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. Signaling Option

The "signaling-option" container captures service-wide attributes of the L2VPN instance.

Although topology discovery or network device configuration is purposely out of scope for the L2SM model, certain VPN parameters for discovery are listed here. The information can then be passed to other elements in the whole automation eco-system (such as the configuration engine) which will handle the actual service provisioning function.

[RFC6074] describes the provisioning, auto-Discovery, and signaling in L2VPNs. It specifies a number of L2VPN provisioning models, and further specifies the semantic structure of the endpoint identifiers required by each model, as well as the distribution of these identifiers by the discovery process, and then specifies how the endpoint identifiers are carried in the signaling protocols (e.g. LDP and L2TPv3).

The "signaling-option" list uses the "type" as the index. The "type" leaf is for the signaling protocol: BGP- L2VPN, BGP-EVPN, T-LDP or L2TP.

5.3.2.1. BGP L2VPN

[RFC4761] and [RFC6624] describe the mechanism to auto-discover L2VPN VPLS/VPWS end points (CE-ID or VE-ID) and signal the label base and offset at the same time to allow remote PE to derive the VPN label to be used when sending packets to the advertising router.

In addition [RFC6624] makes interesting considerations about the L2VPN Scaling scheme and the separation of Administrative Responsibilities between Customer and Service Provider.

Due to the way auto-discovery operates, PEs that have at least one attachment circuit associated with a particular VPN service do not need to be specified explicitly.

In the L2SM model, the target community (or communities) is also not specified since the management system allocates the route target upon receiving VPN creation request.

The "type" leaf under "mp-bgp-l2vpn" is an identityref to specify "vpws" or "vpls" sub-types.

5.3.2.2. BGP EVPN

Defined in [RFC7432], EVPN is an L2VPN technology based upon BGP MAC routing. It provides similar functionality to BGP VPWS/VPLS with improvement around redundancy, multicast optimization, provisioning, and simplicity.

Due to the way auto-discovery operates, PEs that have at least one attachment circuit associated with a particular VPN service do not need to be specified explicitly.

In the L2SM model, the target community (or communities) is also not specified since the management system allocates the route target upon receiving VPN creation request.

The "type" leaf under "mp-bgp-evpn" is an identityref to specify "vpws" or "vpls" sub-types.

5.3.2.3. LDP Pseudowires

[RFC4762] specifies the method of using targeted LDP sessions between PEs to exchange VC label information. This requires a configured full mesh of targeted LDP sessions between all PEs.

As multiple attachment circuits may terminate on a single PE, this PE-to-PE mesh is not a per-site attribute. All PEs related to the L2VPN service will be listed in the "t-ldp-pwe" with associated "vc-id".

The "type" leaf under "mp-bgp-evpn" is an identityref to specify "vpws" ,"vpls", "h-vpls"sub-types. In case of "h-vpls", "qinq" leaf must be specified.

5.3.2.4. PWE Encapsulation Type

Based on [RFC4448], there are two types of Ethernet services: "Port-to-Port Ethernet PW emulation" and "VLAN-to-VLAN Ethernet PW emulation", commonly referred to as Type 5 and Type 4 respectively. This concept applies to both BGP L2VPN VPWS/VPLS and T-LDP signaled PWE implementations.

The "pwe-encapsulation-type" has two types: "ethernet" and "ethernet-vlan". If "signaling-option" is "mp-bgp-l2vpn" or "t-ldp-pwe", then "pwe-encapsulation-type" must be set one of "ethernet" and "ethernet-vlan" .

5.3.2.5. PWE MTU

During the signaling process of a BGP-L2VPN or T-LDP pseudowire, the pwe-mtu value is exchanged and must match at both ends. By default, the pwe-mtu is derived from physical interface MTU of the attachment circuit minus the EoMPLS transport header. In some cases, however, the physical interface on both ends of the circuit might not have identical MTU settings. For example, due to 802.1ad q-in-q operation, an I-NNI will need an extra four bytes to accommodate the S-tag. The inter-carrier E-NNI link may also have a different MTU size than the internal network interfaces.

[RFC4448] requires the same MTU size on physical interfaces at both ends of the pseudowire. In actual implementations, many router vendors have provided a knob to explicitly specify the pwe-mtu, which can then be decoupled from the physical interface MTU.

When there is a mismatch between the physical interface MTU and configured pwe-mtu, the "allow-mtu-mismatch" leaf in the "pwe-mtu" contained enables definition of the required behavior.

5.3.2.6. Control Word

A control word is an optional 4-byte field located between the MPLS label stack and the Layer 2 payload in the pseudowire packet. It plays a crucial role in Any Transport over MPLS (AToM). The 32-bit field carries generic and Layer 2 payload-specific information, including a C-bit which indicates whether the control word will present in the Ethernet over MPLS (EoMPLS) packets. If the C-bit is set to 1, the advertising PE expects the control word to be present in every pseudowire packet on the pseudowire that is being signaled. If the C-bit is set to 0, no control word is expected to be present.

Whether to include control word in the pseudowire packets MUST match on PEs at both ends of the pseudowire and it is non-negotiable during the signaling process.

The use of a control-word applies to pseduowires signaled using either BGP L2VPN VPWS/VPLS or T-LDP. It is a routing-instance level configuration parameter in many cases.

The optional "control-word" leaf is a Boolean field in the L2SM model for the provider to explicitly specify whether the control-word will be signaled for the service instance.

5.3.2.7. L2TP Pseudowires

In the L2VPN framework , a LAC is a Provider Edge (PE) device. In the LAC-LAC reference model, a LAC serves as a cross-connect between attachment circuits and L2TP sessions. Each L2TP session acts as an emulated circuit, also known as pseudowire. A pseudowire is used to bind two attachment circuits together. For different L2VPN applications, different types of attachment circuits are defined.

The "encapsulation-type" has one type,i.e.,l2tp type.

The optional "control-word" leaf is a Boolean field in the L2SM model for the provider to explicitly specify whether the control-word will be signaled for the service instance.

5.3.3. Load Balance Option

As the subscribers start to deploy more 10G or 100G Ethernet equipment in their network, the demand for high bandwidth Ethernet connectivity services increases. These high bandwidth service requests also pose challenges on capacity planning and service delivery in the provider's network, especially when the contractual bandwidth is at, or close to, the speed of physical links of the service provider's core network. Because of the encapsulation overhead, the provider cannot deliver the throughput in the service level agreement over a single link. Although there may be bundled aggregation links between core network elements, or Equal Cost Multiple Paths (ECMP) in the network, an Ethernet-over-MPLS (EoMPLS) PWE or VxLAN circuit is still considered with a single data flow to a router or switch which uses the normal IP five-tuples in the hashing algorithm.

Without burdening the core routers with additional processing of deep inspection into the payload, the service provider now has the option of inserting a flow or entropy label into the EoMPLS frames, or using different source UDP ports in case of VxLAN/EVPN, at ingress PE to

facility load-balancing on the subsequent nodes along the path. The ingress PE is in a unique position to see the actual unencapsulated service frames and identify data flows based on the original Ethernet and IP header.

On the other hand, not all Layer 2 Ethernet VPNs are suited for load-balancing across diverse ECMP paths. For example, a Layer 2 Ethernet service transported over a single RSVP signaled Label Switched Path will not take multiple ECMP routes. Or if the subscriber is concerned about latency/jitter, then diverse path load-balancing can be undesirable.

The optional "load-balance-option" container is used to capture the load-balancing agreement between the subscriber and the provider. If the "load-balance" Boolean leaf is marked TRUE, then one of the following load-balance methods can be selected: "fat-pw", "entropy-label", or "vxlan-source-udp-port". FAT pseudowires are used to load-balance traffic in the core when equal cost multipaths are used. The MPLS labels add an additional label to the stack, called the flow label, which contains the flow information of a VC.

5.3.4. 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 5.2.8.

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.

The type of site-network-access may have an impact on the parameters offered to the customer, e.g., an SP may not offer encryption for multipoint accesses. It is up to the provider to decide what parameter is supported for point-to-point and/or multipoint accesses; which is out of scope for this document. Some containers proposed in the model may require extensions in order to work properly for multipoint accesses.

5.3.4.1. Bearer

The "bearer" container defines the requirements for the site attachment to the provider network that are below Layer 3.

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

5.3.4.2. 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 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" is for user to select between Ethernet encapsulation (port-based) or Ethernet VLAN encapsulation (VLAN-based). All allowed Ethernet interface types of service frames can be listed under "ether-inf-type", e.g., Dot1q interface, physical interface, LAG interface

Corresponding to "ether-inf-type",the connection container also presents three sets of link attributes: dot1q interface,physical interface or optional LAG interface attributes. These parameters are essential for the connection between customer and provider edge devices to establish properly. The connection container also defines L2CP attribute to allow control plane protocol interaction between the CE devices and PE device.

5.3.4.2.1. Dot1q Interface

A VLAN can be configured for VLAN tagged traffic.If the dot1q service is enabled on a logical unit on the connection at the interface, "encapsulation-type ","eth-inf-type" should be specified as Ethernet VLAN ecapsulation(VLAN-based) and Dot1q interface respectively.

In addition, "l2-access-type" should be specified under "dot1q" container to determines how VLAN tagging needs to be done.The current model proposed 5 ways to perform VLAN tagging:

- dot1q: 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 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.

5.3.4.2.2. Physical Interface

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

5.3.4.2.3. 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, 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 PDU will be exchanged between the edge device 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 versus 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 valid path by specifying the optional "mini-link" attribute. To enable fast detection of faulty links, micro-bfd runs independent UDP sessions to monitor the status of each member link. Customer and provider should consent to 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.

5.3.4.2.4. L2CP Control

Customer and Service provider should make pre-arrangement on 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 [802.1D]) can be employed by customers.

To support efficient dynamic transport, Ethernet multicast control frames (e.g., GARP/GMRP [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).

To facilitate interoperability between different Multiple System Operators (MSOs), interaction between the edge device of each administrative domain can be either allowed or keep each Administrative domain control plane separate on a per-protocol basis.

the MEF has provided normative guidance on Layer 2 Control Protocol (L2CP) processing requirements for each service type.

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.

In addition, "provider-bridge-group" and "provider-bridge-mvrp" addresses are also listed in the L2CP container.

5.4. Site Role

A VPN has a particular service topology, as described in Section 5.1.3. As a consequence, each site belonging to a VPN is assigned with 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

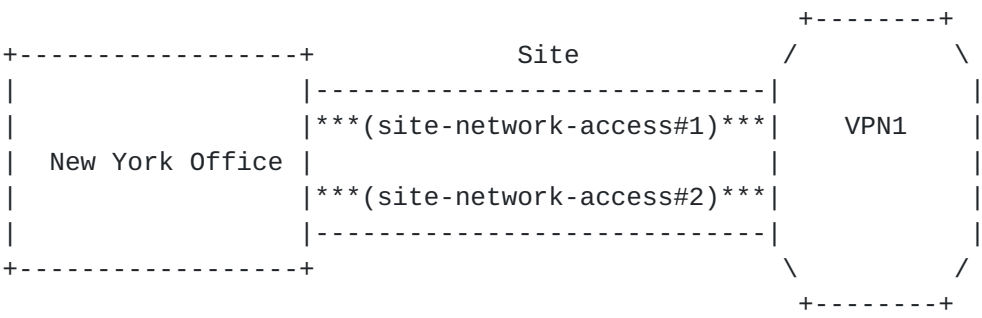
A site may be part of one or multiple VPNs. The "site-type" defines the way the VPN multiplexing is done. There are three possible types of external facing connections associated with an Ethernet VPN

service and a site. Therefore the current version of the model supports three 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-enni: The site represents an ENNI where two Ethernet service providers inter-connect with each other.
- site-vpn-flavor-e2e: The site represents end to end mult-segment connection.

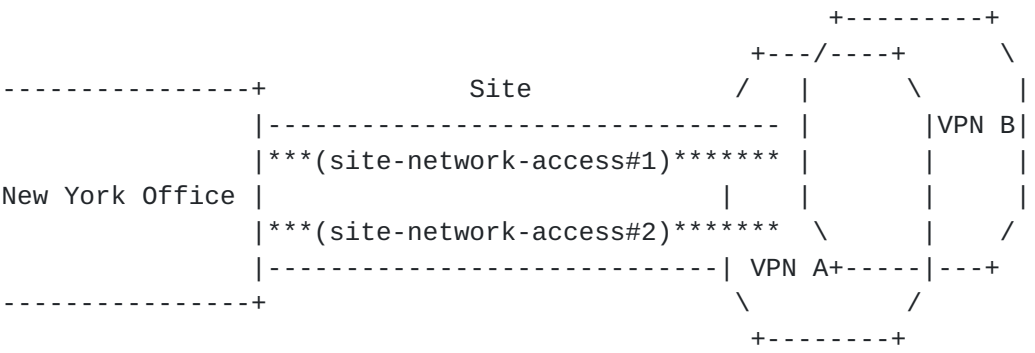
5.5.1.1. Single VPN Attachment: site-vpn-flavor-single

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.

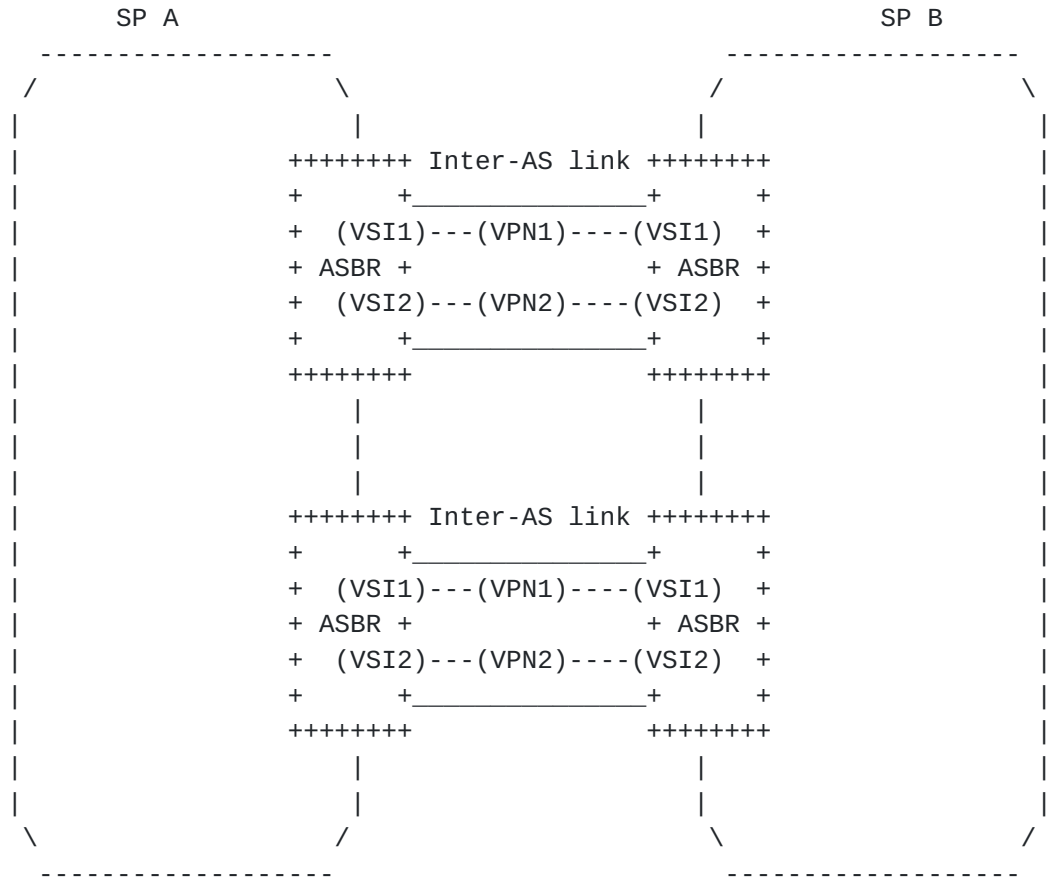


In the example above, the New York office is multihomed. Both logical accesses are using the same VPN attachment rules, and both are connected to VPN A and VPN B.

Reaching VPN A or VPN B from the New York office will be done via destination-based routing. Having the same destination reachable from the two VPNs may cause routing troubles. The customer administration's role in this case would be to ensure the appropriate mapping of its prefixes in each VPN.

5.5.1.3. ENNI: site-vpn-flavor-enni

A External Network-to-Network Interface (ENNI) 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 ENNI, as specific default device configuration parameters may be applied in the case of ENNIs (e.g., ACLs, routing policies).



The figure above describes an option A ENNI 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-enni will be used to inform SP B that this is an ENNI 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 using EVC and OVC container. 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 connectivity, as specific default device configuration parameters may be applied in case of site-vpn-flavor-e2e (e.g., QoS configuration). In order to establish connection between Site 1 in SP A and Site 2 in SP B spanning across multi-domains, SP A may request the creation of end to end connectivity to SP B. The site-vpn-flavor-e2e will be used to inform 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.

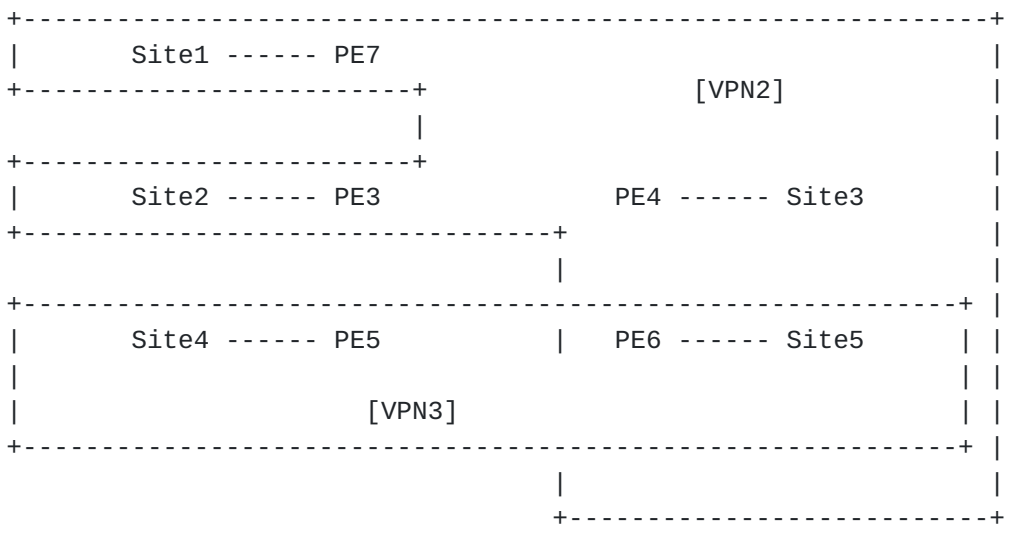
```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>LA1</site-network-access-id>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
  <site-network-access>
   <site-network-access-id>LA2</site-network-access-id>
   <vpn-attachment>
    <vpn-id>VPNB</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
</site>
```

The example above describes a multiVPN 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 multiVPN 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 of the site should be part of a particular 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 multiVPN scenario as follows:

```
      <site>
       <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>
         <site-network-access-id>LA1</site-network-access-id>
         <vpn-attachment>
          <vpn-policy-id>POLICY1</vpn-policy-id>
         </vpn-attachment>
        </site-network-access>
       </site-network-accesses>
      </site>
```

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:

```
      <site>
       <site-id>Site5</site-id>
       <vpn-policies>
        <vpn-policy>
         <vpn-policy-id>POLICY1</vpn-policy-id>
         <entries>
          <id>ENTRY1</id>
          <filter>
           <lan-tag>LAN1</lan-tag>
          </filter>
          <vpn>
           <vpn-id>VPN2</vpn-id>
           <site-role>hub-role</site-role>
          </vpn>
         </entries>
         <entries>
          <id>ENTRY2</id>
          <filter>
           <lan-tag>LAN2</lan-tag>
          </filter>
           <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>
         <site-network-access-id>LA1</site-network-access-id>
         <vpn-attachment>
          <vpn-policy-id>POLICY1</vpn-policy-id>
         </vpn-attachment>
        </site-network-access>
       </site-network-accesses>
      </site>
```

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, aggregation switch).

The current model proposes parameters and constraints that can influence the meshing of the site-network-access.

The management system should honor any customer constraints. If a constraint is too strict and cannot be fulfilled, the management

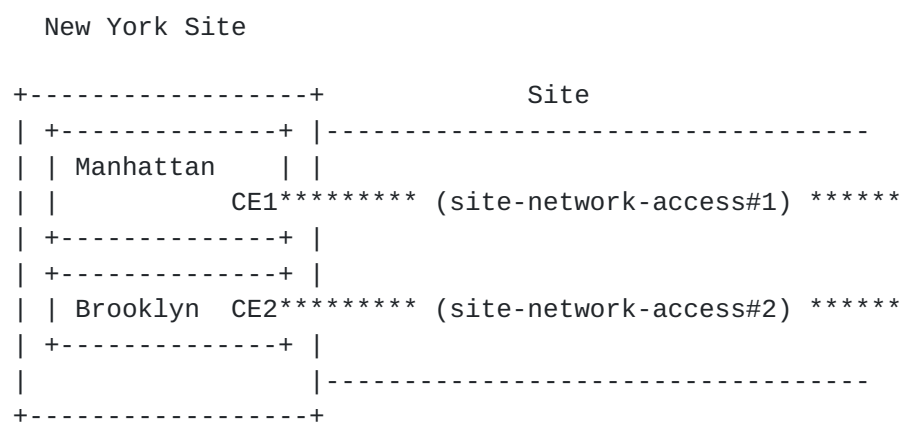
system must not provision the site and should provide relevant information to the user. How the information is provided is out of scope for this document. Whether or not to relax the constraint would then be left up to the user.

Parameters are just hints for the management system for service placement.

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: 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. The customer may force a particular site-network-access to be connected on a particular device that he ordered.



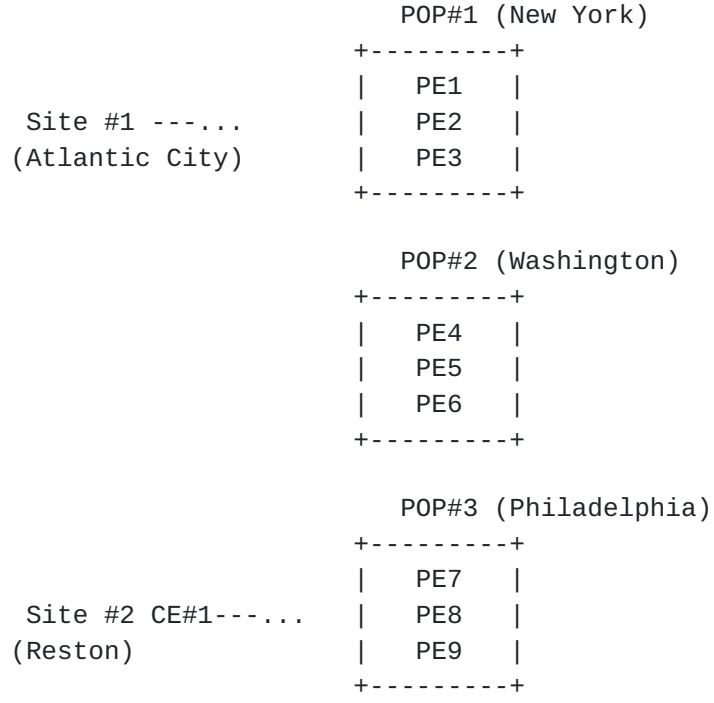
In the figure above, site-network-access#1 is associated with CE1 in the service request. The SP must ensure the provisioning of this connection.

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 ALPHA-2 code.

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 a pure 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 MUST never be taken into account in the targeted set -- 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 proposes 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.

- 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 VSI or MAC-VRF on the target PE and an RD for this VSI or MAC-VRF.This RD MUST be unique across all MAC-VRFs on the target PE.

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

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

The management system MAY apply a per-VPN or per-VSI allocation policy for the RD, depending on the SP's policy. In a per-VPN allocation policy, all VSIs (dispatched on multiple PEs) within a VPN will share the same RD value. In a per-VSI model, all VSIs or 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 5.2.3.1 could be reused in this scenario.

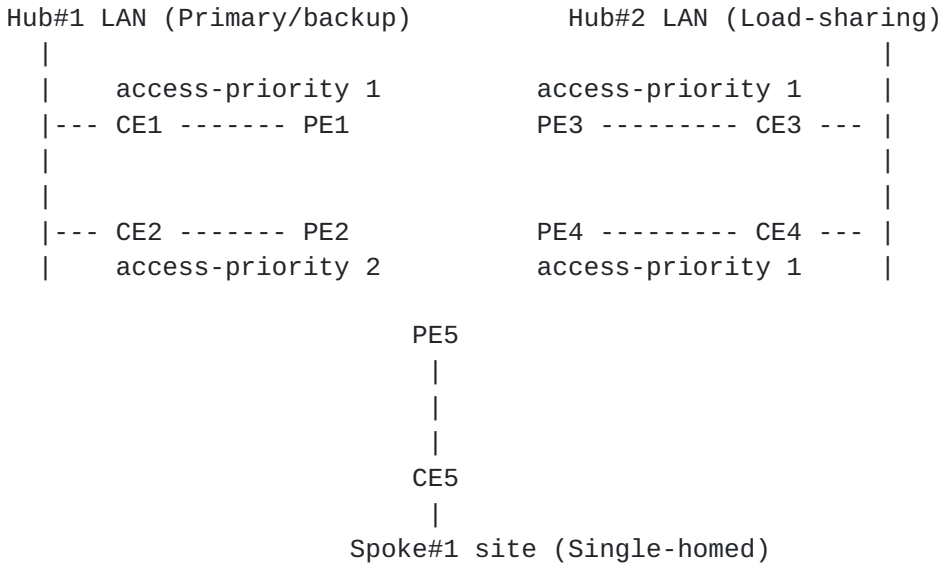
Note that an SP MAY configure a target PE for an automated allocation of RDs. In this case, there will be no need for any backend system to allocate an RD value.

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 be modeled. Let's consider a Hub site with five accesses to the network (A1,A2,A3,A4,A5). The customer wants to load-share its traffic on A1,A2 in the nominal situation. If A1 and A2 fail, the customer wants to load-share its traffic on A3 and A4; finally, if 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 A1 OR A2 is down is not achievable. But the authors believe that using the access-priority attribute will cover most of the deployment use cases and that the model can still 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. The requested bandwidth is expressed as svc-input-bandwidth and svc-output-bandwidth. Input/output direction is using customer site as reference: input bandwidth means download bandwidth for the site, and output 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 input and output bandwidth will allow service provider to know if a customer allows for asymmetric bandwidth access like ADSL. It can also be used to set a rate-limit in a different way for upload and download on symmetric bandwidth access.

The svc-input-bandwidth or svc-output-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-id.

- opaque bandwidth is the total bandwidth that is not associated with any particular cos-id, vpn-id or site network access id.

The svc-input-bandwidth or svc-output-bandwidth must include a "cos-id" parameter if the 'type' is set as 'bw-per-cos'. The cos-id can be assigned based on dot1p value in C-tag, or DSCP in IP header. Ingress service frames are metered against the bandwidth profile based on the cos- identifier.

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

The svc-input-bandwidth or svc-output-bandwidth must include specific "vpn-id" parameter if the 'type' is set as 'bw-per-svc'. Multiple input/output-bandwidth 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 physical port reference or a more specific flow definition (based layer 2 source and destination MAC address, 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 "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. 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.

5.10.2.2. QoS Profile

User can choose between 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 which qos profile is applied to. Our proposed model supports "Site-to-WAN" direction, "WAN-to-Site"direction and "both" direction. By default, "both" direction is used. In case of "both" direction, 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.

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

- rate-limit: Used to rate-limit the class of service. The value is expressed as a percentage of the global service bandwidth. When the qos-profile is implemented at CE side the svc-output-bandwidth is taken into account as reference. When it is implemented at PE side, the svc-input-bandwidth is used.

- frame-delay: Used to define the latency constraint of the class. The latency constraint can be expressed as the lowest possible

latency or a latency boundary expressed in milliseconds. How this latency constraint will be fulfilled is up to the service provider implementation: a strict priority queueing may be used on the access and in the core network, and/or a low latency routing 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 a jitter boundary expressed in microseconds. How this jitter constraint will be fulfilled is up to the service provider implementation: a strict priority queueing may be used on the access and in the core network, and/or a jitter-aware routing 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 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 reservation, controller reservation) is out of scope for this document.

Some constraints may not be offered by an SP; in this case, a deviation should be advertised. 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. Multicast

The "broadcast-unknowunicast-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 CE(s) EXCEPT for the VLAN ID field, ensuring that it is transparently transported. If VLAN IDs are assigned by the SP, they can 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.

B-U-M (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 B-U-M frame disposition rules can be implemented at external facing interfaces (UNI or E-NNI) to rate-limit the B-U-M frames, in term of number of packets per second or bits per second.

The threshold can apply to all B-U-M 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 IP 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. Security

5.12.1. 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.

5.12.2. MAC Address Limit

The service provider may choose to impose a per-attachment circuit "mac-addr-limit" in addition to the service-level MAC limit, and specify the behavior when the limit is exceeded accordingly.

5.13. 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 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" currently presents two types of CFM maintenance association (MA): UP MEP for UNI-N to UNI-N Maintenance Association (MA) and DOWN MEP for UNI-N to UNI-C 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 for use 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.

5.14. External ID References

The service model sometimes refers to external information through identifiers. As an example, to order a 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 REST 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.

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

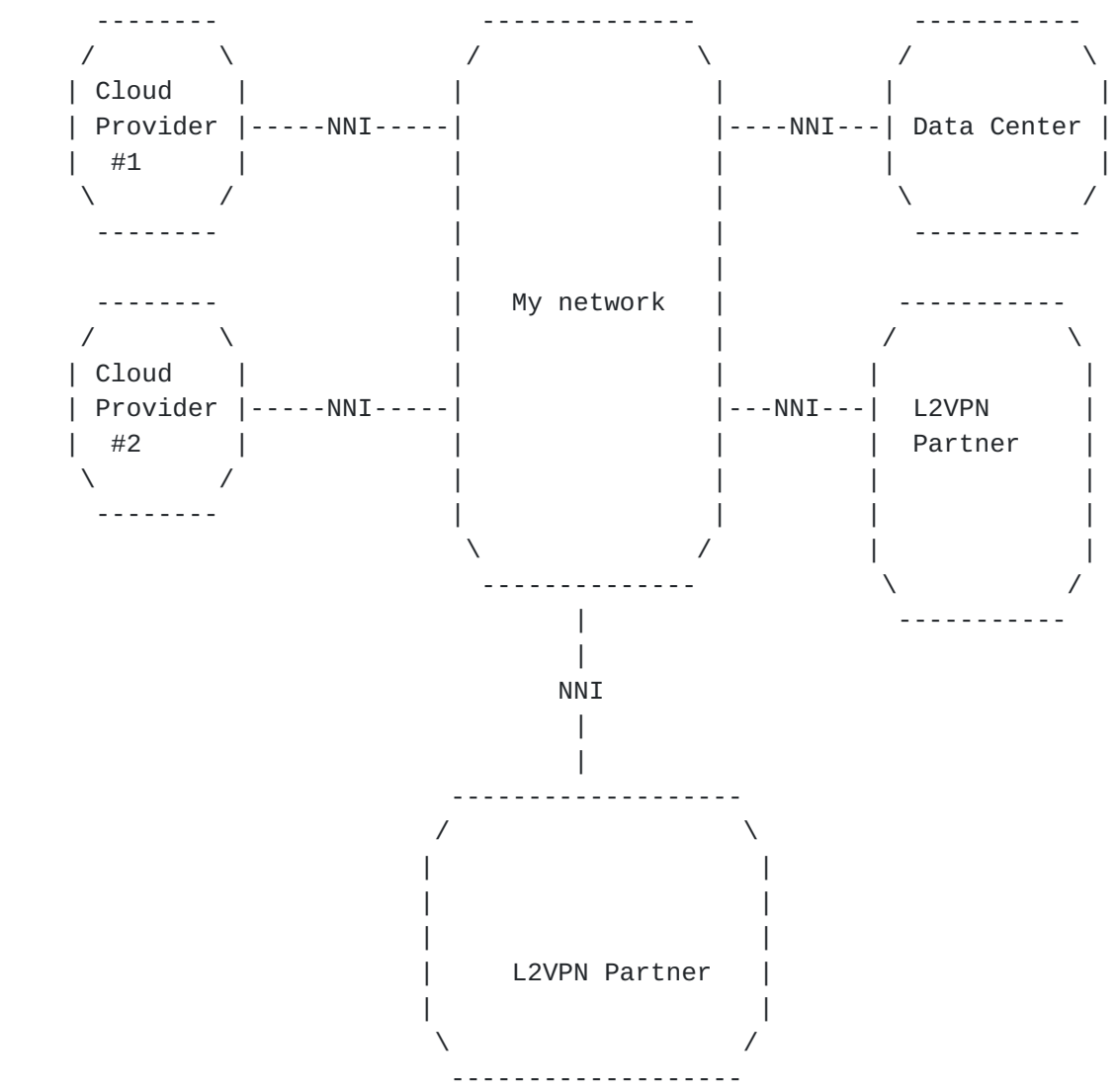
5.15. Defining NNIs and Inter-AS support

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 service seamlessly.

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

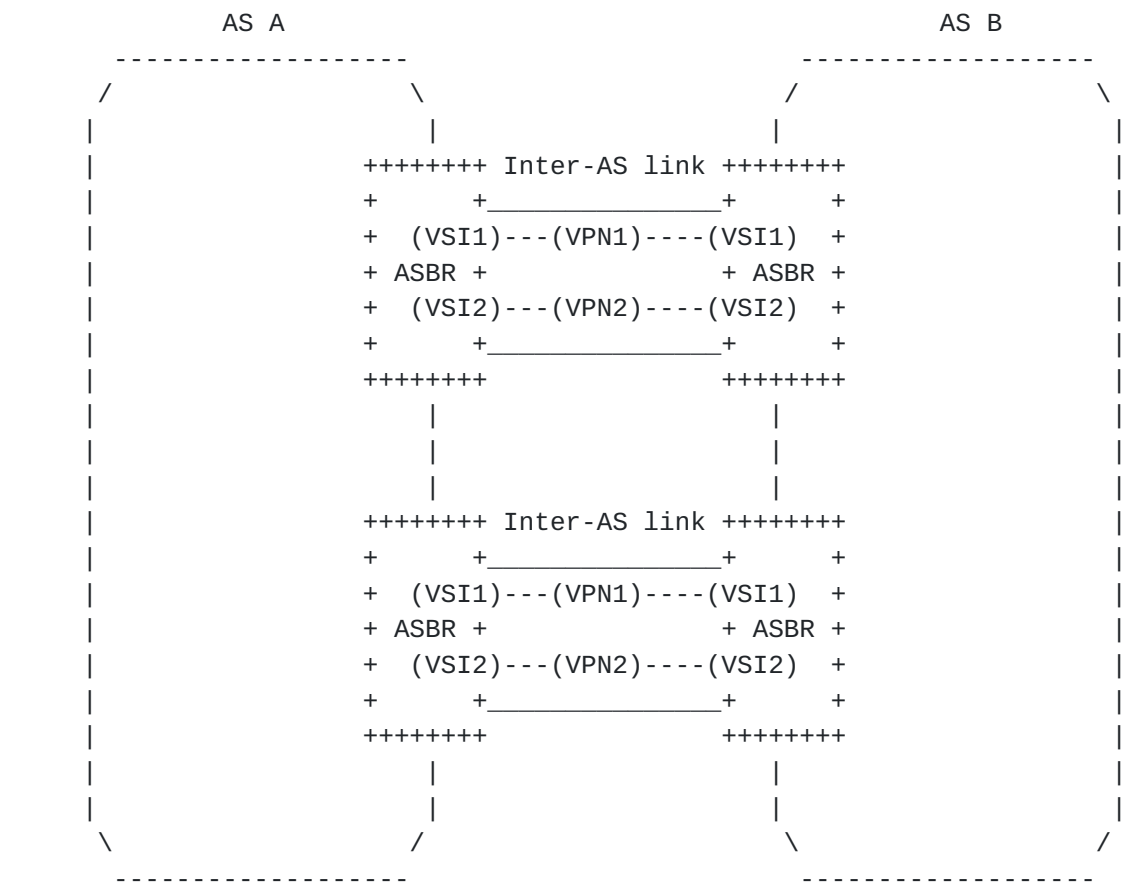
NNIs (network-to-network interfaces) have to be defined to extend the VPNs across multiple ASes. [RFC4761] defines multiple flavors of VPN NNI implementations. Each implementation has pros and cons; this topic is outside the scope of this document. For example, in an Inter-AS option A, autonomous system 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 (VSI) instance and a Border Gateway Protocol (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.

5.15.1. Defining an NNI with the Option A Flavor

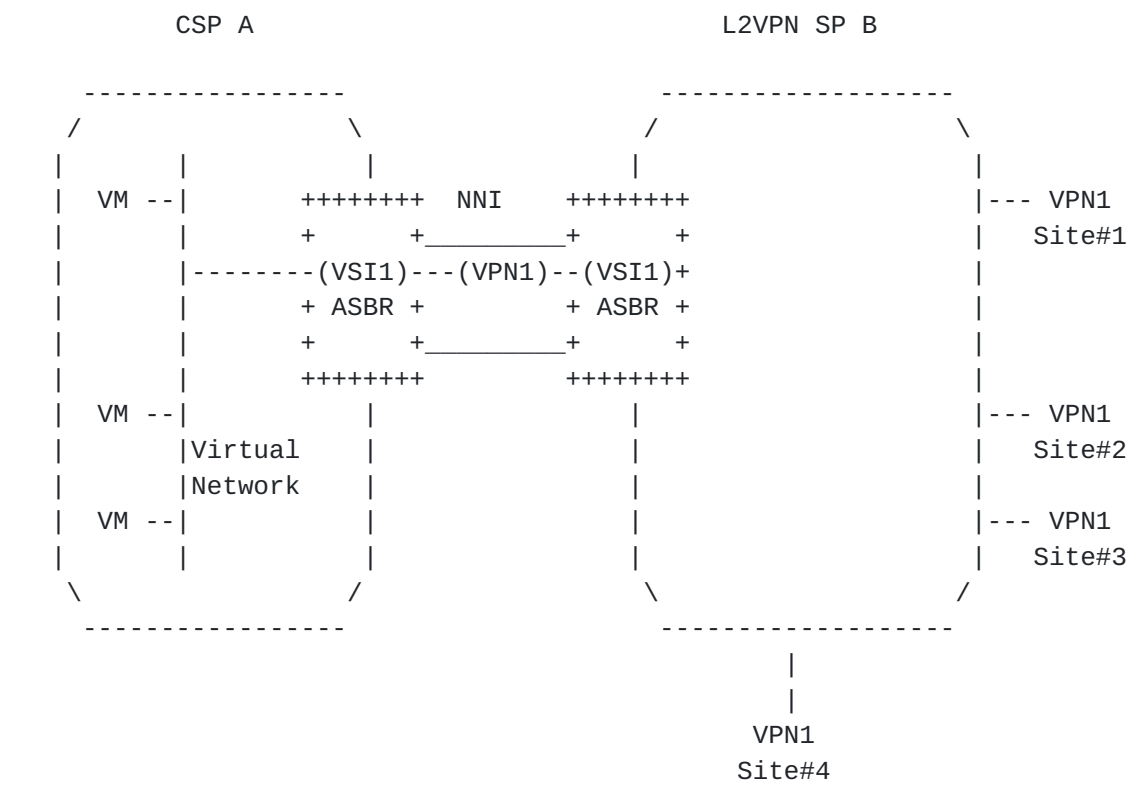


In option A, the two ASes are connected to each other with physical links on ASBRs. For resiliency purposes, there may be multiple physical connections between the ASes. A VPN connection -- physical or logical (on top of physical) -- is created for each VPN that needs to cross the AS boundary, thus providing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



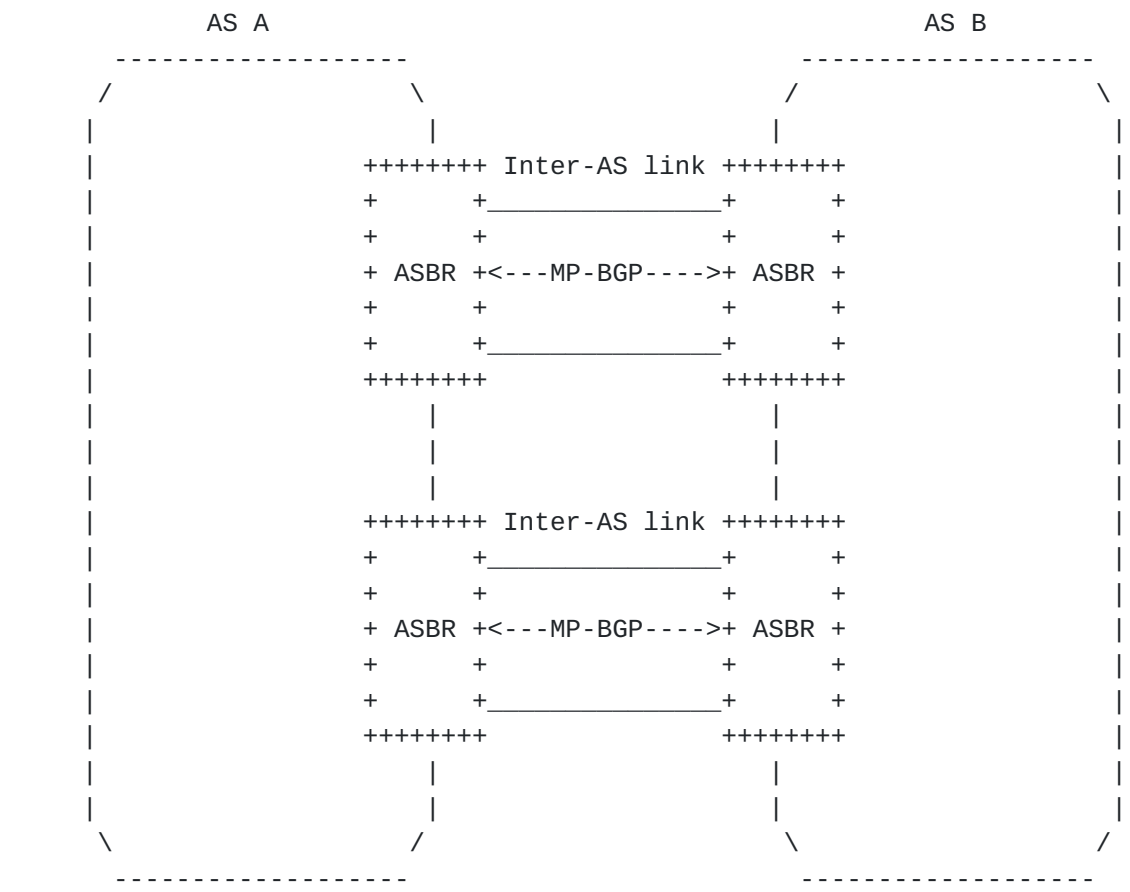
To create the VPN connectivity, the CSP or the customer may use the L2VPN service model that SP B exposes. We could consider that, as the NNI is shared, the physical connection (bearer) between CSP A and SP B already exists. CSP A may request through a service model the creation of a new site with a 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:

```
<site>
    <site-id>CSP_A_attachment</site-id>
    <location>
        <city>NY</city>
        <country-code>US</country-code>
    </location>
    <site-vpn-flavor>site-vpn-flavor-nni</site-vpn-flavor>
    <signaling-options>
        <signaling-option>
           <type>bgp-l2vpn</type>
             <mp-bgp-l2vpn>
                 <svc-id>12456487</svc-id>
                  <type>kompella</type>
              </mp-bgp-l2vpn>
          </signaling-option>
```

```
          <signaling-options>
          <site-network-accesses>
           <site-network-access>
            <site-network-access-id>CSP_A_VN1</site-network-access-id>
                  <connection>
                      <vlan>
                        <vlan-id>17</vlan-id>
                      </vlan>
                      <physical-interface>
                       <encapsulation-type>dot1q</encapsulation-type>
                       </physical-interface>
                    </connection>
                         <service>
                           <svc-ingress-bandwidth>
                             <type>opaque</type>
                             <cir>450000000</cir>
                             <cbs>20000000</cbs>
                             <eir>1000000000</eir>
                             <ebs>200000000</ebs>
                           </svc-ingress-bandwidth>
                           <svc-egress-bandwidth>
                             <cir>350000000</cir>
                             <cbs>10000000</cbs>
                             <eir>800000000</eir>
                             <ebs>200000000</ebs>
                           </svc-egress-bandwidth>
                         </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>
```

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.

5.15.2. 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.

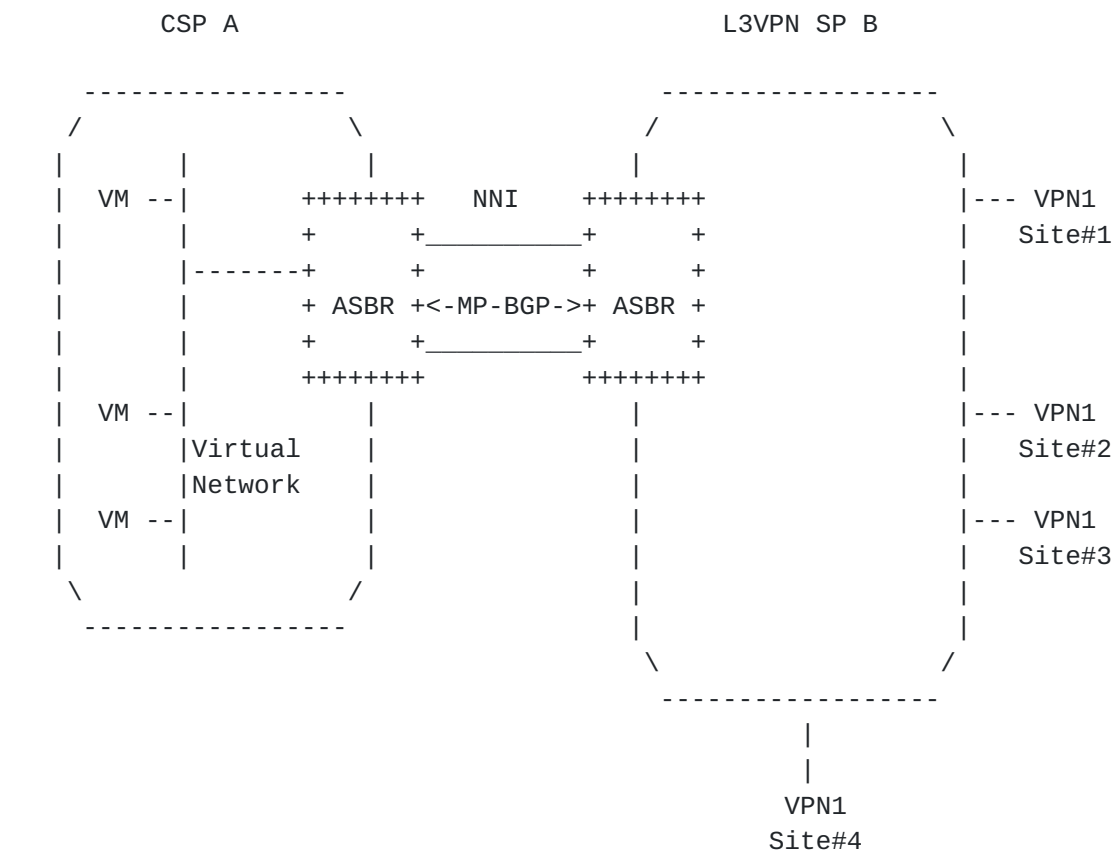
3. The NNI is used between providers that agreed to exchange VPN routes for specific RTs only. Each provider has its own RT scheme. So, a customer spanning the two networks will have different RTs in each network for a particular VPN.

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.

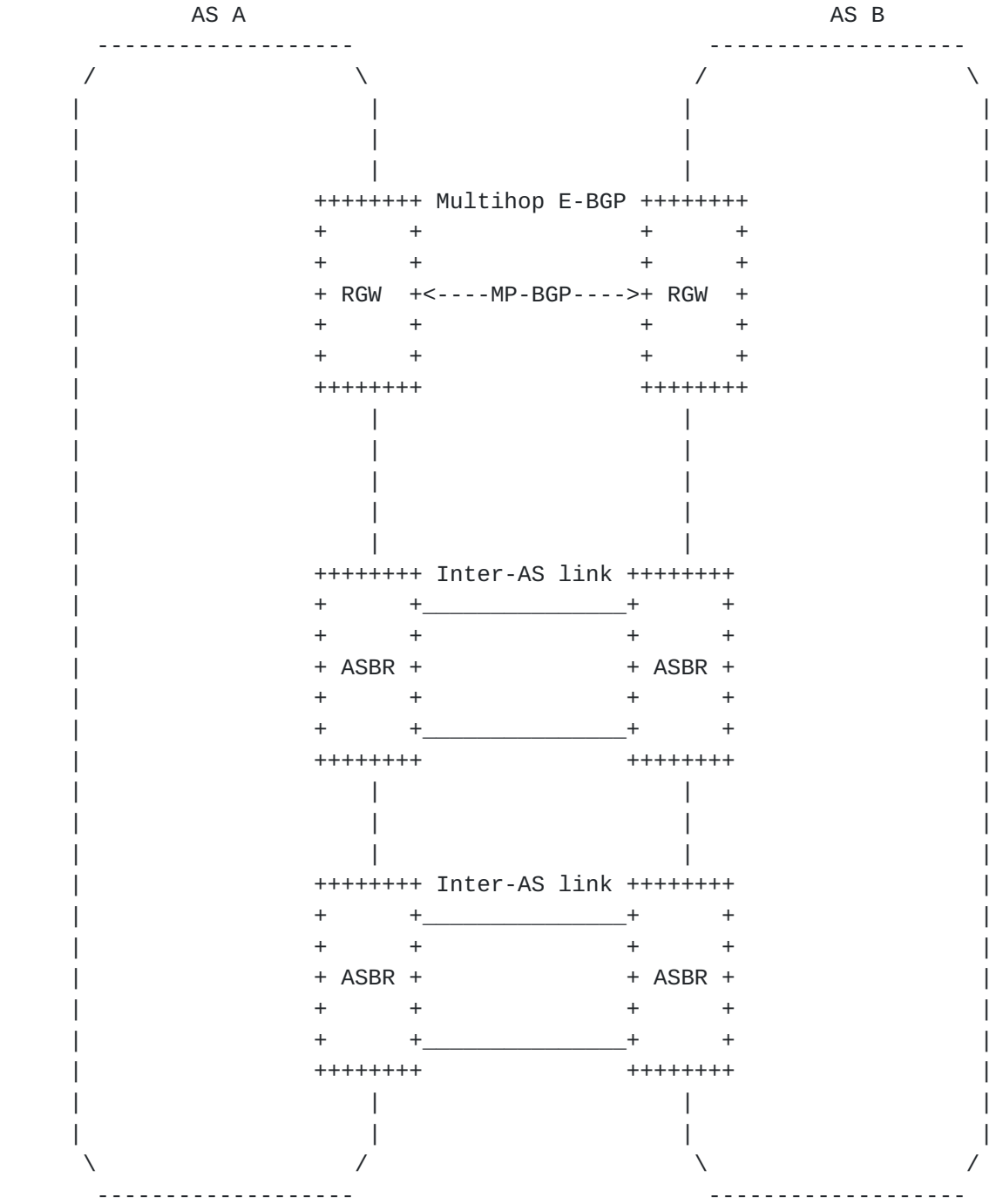
In Case 3, both ASes need to agree on the VPN RT to exchange, as well as how to map a VPN RT from AS A to the corresponding RT in AS B (and vice versa).

Those modelings are currently out of scope for this document.



The example above describes an NNI connection between CSP A and SP network B. Both SPs do not trust themselves and use a different RT allocation policy. 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 virtual network in CSP A to the customer IP VPN (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.

5.15.3. 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.

5.16. 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 approach can be taken:

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

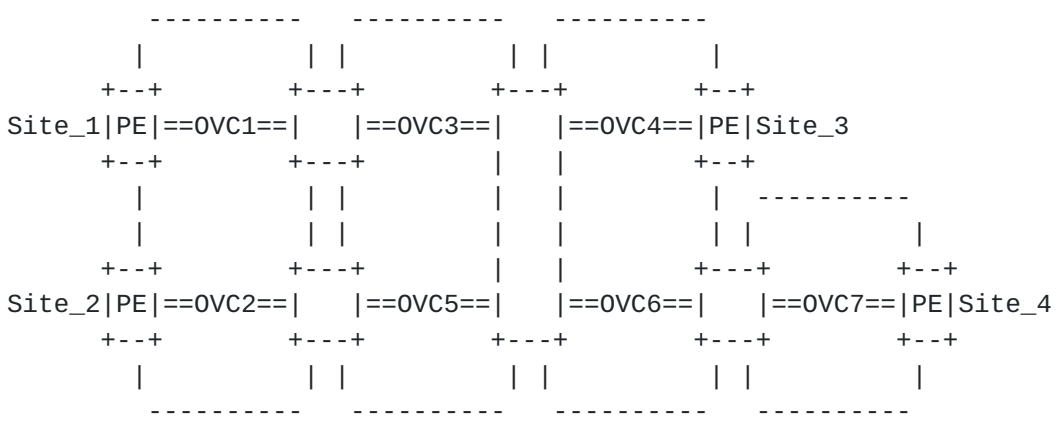
(b) Allow an end-to-end, multi-segment connectivity to be constructed out of several connectivity segments, without maintaining an end-to-end control connection.

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 OVC at different segments, e.g., end to end connectivity between site_1 and Site 3 spans across multiple domains(i.e., Metro networks described in section 5.2.5.) and can be constructed by stitching network access connectivity within site_1 with OVC1, OVC3, OVC4 and network access connectivity within site3 (See the following figure). The assumption is service orchestration layer in figure 5 should have visibility of the complete abstract topology and resource availability. This may rely on network planning to achieve that. The XML below illustrates a possible configuration request to SP:

```
<vpn-service>
<vpn-id>12456487</vpn-id>
     <evc>
         <uni-list>
            <uni>Site1-UNI</uni>
            <uni>Site2-UNI</uni>
            <uni>Site3-UNI</uni>
            <uni>Site4-UNI</uni>
         </uni-list>
      </evc>
      <ovc>
         <ovc-list>
             <ovc>ovc1</ovc>
             <ovc>ovc2</ovc>
             <ovc>ovc3</ovc>
             <ovc>ovc4</ovc>
             <ovc>ovc5</ovc>
```

```
                  <ovc>ovc6</ovc>
                  <ovc>ovc7</ovc>
              </ovc-list>
          </ovc>
      <metro-networks>
              <mero-network>
                 <id>1</id>
                 <inter-mkt-service>TRUE</inter-mkt-service>
                     <intra-mkt>
                         <mkt-name>Metro-Network#1</mkt-name>
                         <ovc>ovc1</ovc>
                     </intra-mkt>
                     <intra-mkt>
                         <mkt-name>Metro-Network#2</mkt-name>
                         <ovc>ovc3</ovc>
                     </intra-mkt>
                     <intra-mkt>
                        <mkt-name>Metro-network#3</mkt-name>
                        <ovc>ovc4</ovc>
                     </intra-mkt>
              </metro-network>
              <mero-network>
                 <id>2</id>
                 <inter-mkt-service>TRUE</inter-mkt-service>
                     <intra-mkt>
                         <mkt-name>Metro-Network#1</mkt-name>
                         <ovc>ovc2</ovc>
                     </intra-mkt>
                     <intra-mkt>
                         <mkt-name>Metro-Network#2</mkt-name>
                         <ovc>ovc5</ovc>
                     </intra-mkt>
                     <intra-mkt>
                        <mkt-name>Metro-network#3</mkt-name>
                        <ovc>ovc6</ovc>
                     </intra-mkt>
                     <intra-mkt>
                        <mkt-name>Metro-network#4</mkt-name>
                        <ovc>ovc7<ovc>
                     </intra-mkt>
               </metro-network>
      </metro-networks>
      </vpn-service>
```

Note that OVC 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 EBGP redistribution of L2VPN NLRI from AS to neighboring AS. First, the PE routers use Internal BGP (IBGP) to redistribute L2VPN NLRI either to an ASBR, or to a route reflector of which an ASBR is a client. The ASBR then uses EBGP 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 Section 3.5.4 and [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 [I-D.ietf-opsawg-service-model-explained], 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 NETCONF/YANG 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:

o Network Instance definition including VPN policy expression.

- Physical interface.
- Ethernet layer (VLAN ID).
- QoS: classification, profiles, etc.
- Signaling Options: support of configuration of all protocols listed in the document, as well as vpn policies associated with these protocols.
- 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 for of a VPN service for 3 sites using point-to-point EVC and a Hub and Spoke VPN service topology as shown in Figure 7. Loadbalancing is not considered in this case.

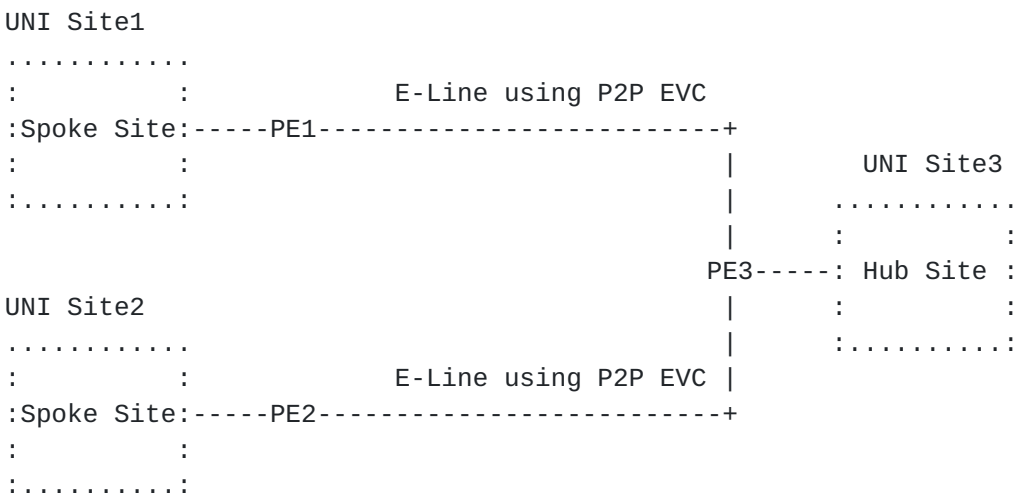


Figure 7: Reference Network for Simple Example

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

```
<vpn-service>
  <vpn-id>12456487</vpn-id>
  <vpn-type>evpl</svc-id-type>
  <evc>
    <uni-list>
      <uni>UNI1</uni>
      <uni>UNI3</uni>
    </uni-list>
  </evc>
  <svc-topo>hub-spoke</svc-topo>
</vpn-service>

<vpn-service>
  <vpn-id>12456488</vpn-id>
  <vpn-type>evpl</svc-id-type >
  <evc>
    <uni-list>
      <uni>UNI2</uni>
      <uni>UNI3</uni>
    </uni-list>
  </evc>
  <svc-topo>hub-spoke</svc-topo>
</vpn-service>
```

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 UNI Site1.

```
<site>
  <site-id>Spoke_Site1</site-id>
  <location>
     <city>NY</city>
     <country-code>US</country-code>
  </location>
  <signaling-options>
    <signaling-option>
      <type>bgp-l2vpn</type>
      <bgp-l2vpn>
         <svc-id>12456487</svc-id>
         <type>kompella</type>
      </mp-bgp-l2vpn>
    </signaling-option>
  </signaling-options>
  <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>
                      <vlan>
                        <vlan-id>17</vlan-id>
                      </vlan>
                      <physical-interface>
                        <encapsulation-type>dot1q</encapsulation-type>
                      </physical-interface>
                    </connection>
                    <service>
                      <svc-ingress-bandwidth>
                        <type>opaque</type>
                        <cir>450000000</cir>
                        <cbs>20000000</cbs>
                        <eir>1000000000</eir>
                        <ebs>200000000</ebs>
                      </svc-ingress-bandwidth>
                      <svc-egress-bandwidth>
                        <cir>350000000</cir>
                        <cbs>10000000</cbs>
                        <eir>800000000</eir>
                        <ebs>200000000</ebs>
                      </svc-egress-bandwidth>
                    </service>
                    <l2cp-protocol>
                      <stp-rstp-mstp>TUNNEL</stp-rstp-mstp>
                      <lldp>TRUE</lldp>
                    </l2cp-protocol>
                    <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>
```

When receiving the request for provisioning 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 SHOULD 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 VFI is already present that fits the needs. If not, it will provision the VFI: Route Distinguisher will come from internal allocation policy model, route-targets are coming 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 vfi 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 VFI 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 vfi one

!
l2 router-id 10.100.1.1
!

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

!
router bgp 1
 bgp log-neighbor-changes
 neighbor 10.100.1.4 remote-as 1
 neighbor 10.100.1.4 update-source Loopback0
 !
 address-family l2vpn vpls
  neighbor 10.100.1.4 activate
  neighbor 10.100.1.4 send-community extended
  neighbor 10.100.1.4 suppress-signaling-protocol ldp
 exit-address-family

!
interface vlan 100 <-- Associating the Attachment
  no ip address       Circuit with the VSI at the PE
  xconnect vfi 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 1.1.1.1 100 encapsulation mpls
!
```

8. YANG Module

```
<CODE BEGINS> file "ietf-l2vpn-svc@2017-08-25.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 iana-if-type {
prefix ianaift;
}
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 2017-08-25{
description
"Initial revision -03 version";
reference
"draft-ietf-l2sm-l2vpn-service-model-03.txt
A YANG Data Model for L2VPN Service Delivery.";
}
/* Features */
feature multicast{
```

```
description
"Enables multicast capabilities in a VPN.";
}

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 evc {
description
"Enable the support of EVC in a VPN.";
}
feature ovc {
description
"Enable the support of OVC in a VPN.";
}
feature cloud-access {
description
"Allow VPN to connect to a Cloud 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 qinq {
description
"Enable the support of QinQ.";

}
feature qinany{
description
"Enable the support of QinAny.";
}
feature vxlan {
description
"Enable the support of VxLAN.";
}
feature ipv4 {
description
"Enables IPv4 support in a VPN.";
}
feature ipv6 {
description
"Enables IPv6 support in a VPN.";
}
feature lan-tag {
description
"Enables LAN Tag support in a VPN.";
}
/* Typedefs */
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 bit 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";
}
enum tunnel {
description
```

```
"Tunnel mode";

}
enum discard {
description
"Discard mode";
}
}
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";
}

/* Identities */
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 pwe-encapsulation-type{
description
"Identity pwe-encapsulation-type";
}

identity ethernet-over-mpls {
base pwe-encapsulation-type;
description
"Identity for ethernet over mpls";
```

```
}
identity ethernet-tagged-mpls {

base pwe-encapsulation-type;
description
"Identity for ethernet tagged over mpls";

}

identity l2tp-pw-type {
description
"Identity for L2TP PW type";
}

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

identity protection-mode {
description
"Identity of protection mode";
}

identity oneplusone{
base protection-mode;
description
"In this scheme, the primary circuitEVC or OVC will be
protected by a backup circuitEVC or OVC, typically meeting certain
diverse path/fiber/site/node criteria.  Both primary and
protection circuits are provisioned to be in the active forwarding
state.  The subscriber may choose to send the same service frames
across both circuits simultaneously.";
}

identity one2one{
base protection-mode;
description
```

```
"In this scheme, a backup circuit to the primary
circuit is provisioned.  Depending on the implementation
agreement, the protection circuits may either always be in active

forwarding state, or may only become active when a faulty state is
detected on the primary circuit.";
}

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

identity phy-inf {
base eth-inf-type;

description
"Identity of Physical Interface type";
}

identity lag-inf {
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-evc-ovc {
base bw-type;
description
"Bandwidth is per evc or per ovc";
}
identity bw-per-port {
base bw-type;
description
"Bandwidth is per site network access";
}

identity opaque {
base bw-type;
```

```
description
"Opaque";
}
identity site-type {

description
"Identity of site type.";
}
identity uni {
base site-type;
description
"Identity of User Network Interface ";
}
identity enni {
base site-type;
description
"Identity of External Network to Network Interface";

}

identity service-type {
description
"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 evpn {
base service-type;
description
"Ethernet VPN Service Type,
Ethernet VPNs are specified in RFC 7432";
}

identity vpls-ldp {
base service-type;
description
"LDP based multipoint Virtual Private LAN services (VPLS) Service Type.
This VPLS uses LDP-signaled Pseudowires";
```

```
}

identity vpls-bgp {
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 epl {
base service-type;
description
"Ethernet Private Line (EPL) Service Type. ";
}
identity evpl {
base service-type;
description

"Ethernet Virtual Private Line (EVPL) Service Type. ";
}

identity ep-lan {
base service-type;
description

" Ethernet Private LAN (EP-LAN)Service Type. ";
}
identity evp-lan {
base service-type;
description
" Ethernet Virtual Private LAN (EVP-LAN)Service Type. ";
}

identity bundling-type {
description
"Bundling type.";
}
identity bundling {
base bundling-type;
description
"Identity for bundling";
}
identity all2one-Bundling {
base bundling-type;
description
"Identity for all to one bundling";
}
```

```
identity color-id {
description

"Identity of color id";
}

identity color-id-evc {
base color-id;
description
"Identity of color id base on EVC";
}
identity color-id-evc-cvlan {
base color-id;
description
"Identity of color id base on EVC and CVLAN ";
}
identity cos-id {
description
"Identity of class of service id";
}

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

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

identity cos-id-ovc-ep {
base cos-id;
description
"Identity of cos id based on OVC EP";
}
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 perf-tier-opt {
description
"Identity of performance tier option.";
}
identity metro {
base perf-tier-opt;

description
"Identity of metro";
}
identity regional {
base perf-tier-opt;
description
"Identity of regional";
}
identity continental {
base perf-tier-opt;

description
"Identity of continental";
}
identity global {
base perf-tier-opt;
description
"Identity of global";
}

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 vpn-signaling-type {
description
"Identity of VPN signaling types";
}
identity bgp-l2vpn {
base vpn-signaling-type;
```

```
description
"Identity of bgp-l2vpn";
}
identity mp-bgp-evpn {
base vpn-signaling-type;
description
"Identity of mp-bgp-evpn";
}

identity t-ldp-pwe3{
base vpn-signaling-type;
description
"Identity of t-ldp-pwe.";
}
identity l2tp-pw {
base vpn-signaling-type;
description
"Identity of l2tp-pw.";
}

identity t-ldp-pwe-type{
description
"Identity for t-ldp-pwe-type.";
}

identity vpws-type {
base t-ldp-pwe-type;

description
"Identity for VPWS";
}

identity vpls-type{
base t-ldp-pwe-type;
description
"Identity for vpls";
}

identity h-vpls{
base t-ldp-pwe-type;
description
"Identity for h-vpls";
}

identity l2vpn-type {
description
"Layer 2 VPN types";
}
```

```
identity kompella {
base l2vpn-type;
description
"Use BGP as signaling protocol.";
}
identity martini {
base l2vpn-type;
description
"Use LDP as signaling protocol";

}
identity both {
base l2vpn-type;
description
"LDP based Signaling and BGP based Auto Discovery";
}

identity evpn-type {
description
"Ethernet VPN types";
}
identity pbb {
base evpn-type;
description
" Provider Backbone Bridging-EVPN";
}

identity management {
description
"Base identity for site management scheme.";
}
identity co-managed {
base management;
description
"Base identity for co-managed site.";
}
identity customer-managed {
base management;
description
"Base identity for customer managed site.";
}

identity provider-managed {
base management;
description
"Base identity for provider managed site.";
}
identity address-family {
```

```
description
"Base 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 IPVPN.";
}

identity spoke-role {
base site-role;
```

```
description

"Spoke Site in a Hub &amp;amp; Spoke IPVPN.";
}
identity hub-role {
base site-role;
description
"Hub Site in a Hub &amp;amp Spoke IPVPN.";
}
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 data-svc-frame-delivery {
description
"Delivery types";

}
identity discard {
base data-svc-frame-delivery;
description
"Service Frames are discarded.";
}
identity unconditional {
base data-svc-frame-delivery;
description
"Service Frames are unconditionally";
}

identity conditional {
base data-svc-frame-delivery;
description
"Service Frame are conditionally
delivered to the destination UNI.";
}
identity evc-type {
description
"Service topology Type";
}
identity point-to-point {
base evc-type;
description
"Point to Point.";
}
```

```
identity multipoint-to-multipoint {
base evc-type;
description
"Multipoint to Multipoint.";
}

identity rooted-multipoint {
base evc-type;
description
"Rooted Multipoint.";
}

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 l2-access-type {
description
"This identify the access type
of the vpn acccess interface";
}
identity untag {
base l2-access-type;
description
"Untag";

}
identity port {
base l2-access-type;
description
"Port";
}
identity dot1q {
base l2-access-type;
description
"Qot1q";
}

identity qinq {
base l2-access-type;
description
"QinQ";
}
identity sub-interface {
base l2-access-type;
description
"Create a default sub-interface and keep vlan";

}
identity vxlan {
base l2-access-type;
description
"Vxlan access into the vpn";
}
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 load-balance-method {
  description
  "Base identity for load balance method.";
}
identity fat-pw {
   base load-balance-method;
   description
```

```
   "Identity for Fat PW. Fat label is
   applied to Pseudowires across MPLS
   network.";
}
identity entropy-label {
   base load-balance-method;
   description
   "Identity for entropy label.Entropy label
   is applied to IP forwarding,
   L2VPN or L3VPN across MPLS network";
}
identity vxlan-source-port {
   base load-balance-method;
   description
   "Identity for vxlan source port.VxLAN
   Source Port is one load balancing method.";

}
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 bidirection {
  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.";
}

/* Groupings */
grouping vpn-service-cloud-access {
container cloud-accesses {
if-feature cloud-access;
list cloud-access {
key cloud-identifier;
leaf cloud-identifier {
 type string;
 description
   "Identification of cloud service. Local
   admin 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.";
```

```
}
  container authorized-sites {
   list authorized-site {
        key site-id;

        leaf site-id {
         type leafref {
          path "/l2vpn-svc/sites/site/site-id";
         }
         description
          "Site ID.";
        }
        description
         "List of authorized sites.";
   }
   description
        "Configuration of authorized sites.";
  }

  container denied-sites {
   list denied-site {
        key site-id;

        leaf site-id {
         type leafref {
          path "/l2vpn-svc/sites/site/site-id";
         }
         description
          "Site ID.";
        }
        description
         "List of denied sites.";
   }
   description
        "Configuration of denied sites.";
  }

description
 "Cloud access configuration.";
}
description
"Container for cloud access configurations";
}
description
"Grouping for vpn cloud definition";
}

grouping site-device {
```

```
container device {
list devices {
key "device-id";
leaf device-id {
 type string;
 description
   "Device ID";
}

leaf location {
 type leafref {
path "/l2vpn-svc/sites/site/locations/location/location-id";
}
 description
   "Site name";
}
container management {
 leaf address {
   type inet:ip-address;

   description
   "Address";
 }
 leaf management-transport {
   type identityref {
   base address-family;
   }
   description
   "Transport protocol used for management.";
 }
 description
   "Container for management";
}
description
 "List of devices";
}
description
"Devices configuration";
}
description
"Device parameters for the site.";
}

grouping site-management {
container management {
leaf type {
type identityref {
```

```
 base management;
}
description
 "Management type of the connection.";
}
description
"Container for management";
}
description
"Grouping for management";

}

grouping site-vpn-policy {
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. ";
}
leaf-list ipv4-lan-prefix {
when "derived-from-or-self(../type, 'l2vpn-svc:ipv4')" {
description
 "Only applies when VPN Policy filter is IPv4 Prefix filter.";
}
 if-feature ipv4;
 type inet:ipv4-prefix;
 description
  "List of IPv4 prefixes as LAN Prefixes to be matched.";
}
leaf-list ipv6-lan-prefix {
when "derived-from-or-self(../type, 'l2vpn-svc:ipv6')" {
description
 "Only applies when VPN Policy filter is IPv6 Prefix filter.";
}
 if-feature ipv6;
 type inet:ipv6-prefix;
 description
  "List of IPv6 prefixes as LAN prefixes to be matched.";
}
 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-svc/vpn-id";
}
mandatory true;
description
"Reference to an IP VPN.";
```

```
}
leaf site-role {
type identityref {
base site-role;
}
default any-to-any-role;
description
"Role of the site in the IP VPN.";
}
description
"List of VPNs the LAN is associated with.";
}
 description
   "List of entries for export policy.";
}
description
 "List of VPN policies.";
}
description
"VPN policy.";
}

description
"VPN policy parameters for the site.";
}

grouping umb-frame-delivery {
leaf unicast-frame-delivery {
type identityref {
base data-svc-frame-delivery;
}
description
"Unicast Data Service Frame Delivery Mode
(unconditional[default], conditional, or discard).";

}
leaf multicast-frame-delivery {
type identityref {
base data-svc-frame-delivery;
}

description

"Multicast Data Service Frame Delivery Mode
(unconditional[default], conditional, or discard).";
}
leaf broadcast-frame-delivery {
type identityref {
```

```
base data-svc-frame-delivery;
}
description
"Broadcast Data Service Frame Delivery Mode
(unconditional[default], conditional, or discard).";
}
description
"Grouping for unicast, mulitcast, broadcast frame delivery";
}

grouping customer-location-info {
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 zip-code {
type string;
description
 "ZIP code of the site.";
}
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.";
}
description
"List for location";
}
description
"Location of the site.";
}
description
"This grouping defines customer location parameters";
}

grouping site-diversity {
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.";
}
description

"This grouping defines site diversity parameters";
}

grouping site-service {
list cvlan-id-to-svc-map {
key "svc-id type";
leaf svc-id {
type leafref {

  path "/l2vpn-svc/vpn-services/vpn-svc/vpn-id";
}
```

```
description
"VPN Service identifier";
}
leaf type {
type identityref {
  base bundling-type;
}
description
  "Bundling type";
}
list cvlan-id {
key vid;
leaf vid {
  type identityref {
  base ianaift:iana-interface-type;
  }
  description
  "CVLAN ID";
}
description
  "List of CVLAN-ID to EVC Map configurations";
}
description
"List for cvlan-id to evc map configurations";
}

description

"This grouping defines site service parameters";
}
grouping service-protection {
container service-protection {
leaf protection-mode {
type identityref {
 base protection-mode;
  }

  description
  "Container of protection mode configurations";
}
description
"Container of End-to-end Service Protection
 configurations";
}
description
"Grouping for service protection";
}
```

```
grouping vpn-service-multicast {
container multicast {
if-feature multicast;
leaf enabled {
type boolean;
default false;
description
"Enables multicast.";
}
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.";
}
leaf traffic-type {
type identityref {
base tf-type;
}
description
"Traffic Type";
}
leaf group-port-mapping {

type identityref {
base mapping-type;
}
description
"Describe the way in which each interface is associated with the Multicast
group";
}
description
"Multicast global parameters for the VPN service.";

}
description
"Grouping for multicast VPN definition.";
}
grouping vpn-extranet {
container extranet-vpns {
if-feature extranet-vpn;
list extranet-vpn {
key vpn-id;
```

```
leaf vpn-id {
type svc-id;
description

"Identifies the target VPN.";
}
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.";
}
description
"List of extranet VPNs the local VPN is attached to.";
}
description
"Container for extranet VPN configuration.";
}
description
"Grouping for extranet VPN configuration.
This provides an easy way to interconnect
all sites from two VPNs.";
}

grouping signaling-options-grouping {
list signaling-options {
key "type";
leaf type {
type identityref {
  base vpn-signaling-type;
}
description
  "VPN signaling types";

}
container bgp-l2vpn {
when "../type = 'bgp-l2vpn'" {
description
"Only applies when vpn signaling type is bgp-l2vpn.";
}
leaf vpn-id {
  type leafref{
  path "/l2vpn-svc/vpn-services/vpn-svc/vpn-id";
 }
```

```
  description
  "Identifies the target VPN";
}
leaf type {
  type identityref {
  base l2vpn-type;
  }
  description
  "L2VPN types";
}
leaf pwe-encapsulation-type {
 type identityref {
 base pwe-encapsulation-type;
}
 description
  "Leaf for PWE Encapsulation Type configurations";
 }
container pwe-mtu {
 leaf allow-mtu-mismatch {
  type boolean;
  description
  "Allow MTU mismatch";
 }
 description
  "Container of PWE MTU configurations";
}
leaf address-family {
 type identityref {
   base address-family;
 }
 description
   "Address family used for management.";
}
description
  "Container for MP BGP L2VPN";
}
container mp-bgp-evpn {
when "../type = 'mp-bgp-evpn'" {

description
"Only applies when vpn signaling type is mp-bgp-evpn.";
}
leaf vpn-id {
  type leafref{
 path "/l2vpn-svc/vpn-services/vpn-svc/vpn-id";
}
description
"Identifies the target EVPN";
```

```
}
leaf type {
  type identityref {
  base evpn-type;
  }
  description
  "L2VPN types";
}
leaf address-family {
 type identityref {
   base address-family;
 }
 description
   "Address family used for management.";
}
leaf mac-learning-mode {
  type identityref {
  base mac-learning-mode;
  }
  description
  "Indicates through which plane MAC addresses are
  advertised.";
}
leaf arp-suppress {
  type boolean;
  default false;
  description
  "Indicates whether to suppress ARP broadcast.";
}
description
  "Container for MP BGP L2VPN";
}
container t-ldp-pwe {
when "../type = 't-ldp-pwe3'" {
description
"Only applies when vpn signaling type is bgp-l2vpn.";
}
leaf type {
type identityref {

base t-ldp-pwe-type;
}
description
"T-LDP PWE type";
}
leaf pwe-encapsulation-type {
 type identityref {
 base pwe-encapsulation-type;
```

```
}
 description
  "Leaf for PWE Encapsulation Type configurations";
 }
container pwe-mtu {
 leaf allow-mtu-mismatch {
  type boolean;
  description
  "Allow MTU mismatch";
 }
 description
  "Container of PWE MTU configurations";
}
list pe-eg-list {
  key "service-ip-addr vc-id";
  leaf service-ip-addr {
  type inet:ip-address;
  description
  "Service ip lo address";
  }
  leaf vc-id {
  type string;
  description
  "VC id";
  }
  leaf peer-id {
  type string;
  description
  "Local Peer id";
  }
  leaf remote-peer-id {
  type string;
  description
  "Remote Peer id";
  }
  leaf pw-priority {
  type uint32;
  description
  "PW priority";
  }
  description
  "List of PE/EG";
}
leaf control-word {
  type boolean;
   description
   "Control word configurations";
}
```

```
container qinq {
when "../type = 'h-vpls'" {
description
"Only applies when t-ldp pwe type is h-vpls.";
}
leaf s-tag {
type uint32;

description
"S-TAG";
}
leaf c-tag {
type uint32;
description
"C-TAG";
}
description
"Container for QinQ";
}
description
"Container of T-LDP PWE configurations";
}

container l2tp-pw{
leaf encapsulation-type {
type identityref {
base encapsulation-type;
}
description
"Encapsulation type";
}
leaf control-word {
  type boolean;
   description
   "Control word configurations";
}
description
"Container for l2tp pw";
}
description
"List of VPN Signaling Option.";
}
description
"Grouping for signaling option";
}

grouping load-balance-grouping {
leaf enable {
```

```
type boolean;
description
"Enable load balancing";
}

leaf load-balance-method {
type identityref {
base load-balance-method;

}
description
"select load balancing method such as
fat-pw, entropy-label, or
vxlan-source-udp-port.";
}
description
"Grouping for load balance ";
}

grouping operational-requirements-ops {
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";
}
description
"This grouping defines some operational parameters
parameters";
}

grouping intra-mkt-grouping {
list intra-mkt {
key "mkt-name";
leaf mkt-name {
type string;
description
```

```
 "MKT Name";

}
leaf ovc-id {

type leafref {
path "/l2vpn-svc/vpn-services/vpn-svc/ovc/ovc-list/ovc-id";
}
description
"OVC id";
}
leaf site-id {
type leafref{
path "/l2vpn-svc/sites/site/site-id";
}
description

"OVC identifier";
}
description
"List of intra-MKT";
}
description
"Grouping for intra-MKT";
}

grouping inter-mkt-service {
leaf inter-mkt-service{
type boolean;
description
"Indicate whether service is inter market service.";
}
description

"Grouping for inter-MKT service";
}
grouping svc-type-grouping {
container evc {
if-feature evc;
leaf enabled {
type boolean;
description
"Enabled EVC";
}
leaf evc-type {
type identityref {
base evc-type;
}
```

```
description
"EVC type";
}

leaf number-of-pe {

type uint32;
config false;
description
 "Number of PE";
}
leaf number-of-site {
type uint32;
config false;
description
 "Number of Sites";
}
container uni-list {
if-feature uni-list;
list uni-list {
 key "uni-site-id";
 leaf uni-site-id {
   type string;
   description
  "UNI site Identifier";
 }
leaf off-net {
type boolean;
description
 "If Off net is set to true, off-net is enabled, if
 off net is set to false, on-net is enabled";
}
leaf service-multiplexing {
 type boolean;
 description
   "If the Service Multiplexing attribute is enabled then
    multiple Ethernet Services can terminate at the UNI.
   If Service Multiplexing is disabled, then only one
   Ethernet Service can terminate at the UNI.";
}
leaf ce-vlan-preservation {
type boolean;
description
"CE vlan preservation";
}

leaf ce-vlan-cos-perservation {
type boolean;
```

```
description
"CE vlan COS preservation";
}
 description
   "List for UNIs";
}
description
 "Container for UNI List";
}

description
"Container for Ethernet virtual connection.";
}
container ovc {
//if-feature ovc;
list ovc-list {
key ovc-id;
leaf ovc-id {
 type svc-id;
 description
   "OVC ID type";
}

leaf off-net {
 type boolean;
 description
   "Off net";
}
leaf svlan-cos-preservation {
type boolean;
description
"SVLAN CoS preservation";

}
leaf svlan-id-preservation {
type boolean;
description
"SVLAN ID preservation";
}
leaf svlan-id-ethernet-tag {
type string;
 description
 "SVLAN-ID/Ethernet Tag configurations";
}
leaf ovc-endpoint {

type string;
description
```

```
"OVC Endpoint";
}
description
 "List for OVC";
}

description
"Container for OVC";
}
description
"Grouping of service types.";
}

grouping cfm-802-grouping {
leaf maid {
type string;
description
"MA ID";
}
leaf mep-id {
type uint32;
description
"Local MEP ID";
}
leaf mep-level {
type uint32;
description
"MEP level";
}
leaf mep-up-down {
type enumeration {
enum up {
 description
   "MEP up";

}
enum down {
 description
   "MEP down";
}
}
description
"MEP up/down";
}
leaf remote-mep-id {
type uint32;
description
"Remote MEP ID";
```

```
}
leaf cos-for-cfm-pdus {
type uint32;
description
"COS for CFM PDUs";

}
leaf ccm-interval {
type uint32;
description
"CCM interval";
}
leaf ccm-holdtime {
type uint32;
description
"CCM hold time";
}
leaf alarm-priority-defect {
type identityref {
base fault-alarm-defect-type;
}
description
"The lowest priority defect that is
allowed to generate a Fault Alarm.
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";
}
description
"Grouping for 802.1ag CFM attribute";
}

grouping y-1731 {
list y-1731 {
key maid;
leaf maid {
type string;
description
 "MA ID ";
}
leaf mep-id {
type uint32;
description
 "Local MEP ID";
```

```
}

leaf type {
type identityref {
 base pm-type;
}

description
 "Performance monitor types";
}
leaf remote-mep-id {
type uint32;
description
 "Remote MEP ID";
}
leaf message-period {
type uint32;
description
 "Defines the interval between OAM messages. The message
  period is expressed in milliseconds";
}
leaf measurement-interval {
type uint32;
description
 "Specifies the measurement interval for statistics. The
  measurement interval is expressed in seconds";
}
leaf cos {
type uint32;
description
 "Class of service";
}
leaf loss-measurement {
type boolean;
description
 "Whether enable loss measurement";
}

leaf synthethic-loss-measurement {
type boolean;
description
 "Indicate whether enable synthetic loss measurement";
}
container delay-measurement {
leaf enable-dm {
 type boolean;
 description
   "Whether to enable delay measurement";
```

```
}
leaf two-way {
 type boolean;
 description

   "Whether delay measurement is two-way (true) of one-
   way (false)";

}
description
 "Container for delay measurement";
}
leaf frame-size {
type uint32;
description
 "Frame size";
}
leaf session-type {
type enumeration {
 enum proactive {
   description
   "Proactive mode";
 }
 enum on-demand {
   description
   "On demand mode";
 }
}
description
 "Session type";
}
description
"List for y-1731.";
}
description
"Grouping for y.1731";
}

grouping enni-site-info-grouping {

container site-info {
leaf site-name {
type string;
description
 "Site name";
}
leaf address {
type inet:ip-address;
```

```
description
 "Address";
}
leaf Edge-Gateway-Device-Info {
type string;
description
 "Edge Gateway Device Info ";

}
description
"Container of site info configurations";
}
description
"Grouping for site information";
}

grouping site-security {
container security {
uses mac-loop-prevention-grouping;
container access-control-list {
list mac {
 key "mac-address";
 leaf mac-address {
   type yang:mac-address;
   description
   "MAC address";
 }
 description
   "List for MAC";
}
description
 "Container for access control";
}
uses mac-addr-limit-grouping;
description
"Security parameters";
}
description
"This grouping defines security parameters for a site";
}

grouping lacp-grouping {
container lacp {
leaf lacp-state {
type boolean;
description
 "LACP on/off";
}
```

```
leaf lacp-mode {
type boolean;
description
 "LACP mode";
}
leaf lacp-speed {
type boolean;
description

 "LACP speed";
}
leaf mini-link {
type uint32;
description
 "The minimum aggregate bandwidth for a LAG";
}
leaf system-priority {
type uint16;
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 micro-bfd-on-off {
 type enumeration {
   enum on {
   description
    "Micro-bfd on";
   }
   enum off {
  description
   "Micro-bfd off";
   }
 }
 description
   "Micro BFD ON/OFF";
}
leaf bfd-interval {
 type uint32;

 description
   "BFD interval";
}
leaf bfd-hold-timer {
 type uint32;
 description
```

```
   "BFD hold timer";
}
description
 "Container of Micro-BFD configurations";
}
container bfd {
if-feature bfd;
leaf bfd-enabled {
 type boolean;

 description

   "BFD activation";
}
choice holdtime {
 case profile {
   leaf profile-name {
   type string;
   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-link-list {
list member-link {
 key "name";
 leaf name {
   type string;

   description
  "Member link name";
 }
 leaf port-speed {
```

```
   type uint32;
   description
   "Port speed";
 }
 leaf mode {
   type neg-mode;
   description
  "Negotiation mode";
 }
 leaf mtu {

   type uint32;
   description
   "MTU";

 }
 container oam-802.3ah-link {
   if-feature oam-3ah;
   leaf enable {
   type boolean;
   description
   "Indicate whether support oam 802.3 ah link";
   }
   description
   "Container for oam 802.3 ah link.";
 }
 description
   "Member link";
}
description
 "Container of Member link list";
}
leaf flow-control {
type string;
description
 "Flow control";
}

leaf lldp {
type boolean;
description
  "LLDP";
}
description
"LACP";

}
description
```

```
"Grouping for lacp";
}

grouping phy-interface-grouping {
container phy-interface {
leaf port-number {
type uint32;
description
 "Port number";
}
leaf port-speed {

type uint32;
description
 "Port speed";
}
leaf mode {
type neg-mode;
description
 "Negotiation mode";
}

leaf phy-mtu {
type uint32;
description
 "PHY MTU";
}
leaf flow-control {
type string;
description
 "Flow control";
}
leaf physical-if {
 type string;
 description
  "Physical interface";
}
leaf circuit-id {
   type string;
   description
 "Circuit ID";
}
leaf lldp {
type boolean;
description
 "LLDP";
}
```

```
container oam-802.3ah-link {
if-feature oam-3ah;
leaf enable {
type boolean;
description
  "Indicate whether support oam 802.3 ah link";
}
description
"Container for oam 802.3 ah link.";
}
leaf uni-loop-prevention {
type boolean;

description
 "If this leaf set to truth that the port automatically
  goes down when a physical loopback is detect.";
}
description
"Container of PHY Interface Attributes configurations";
}
description
"Grouping for phy interface.";
}

grouping lag-interface-grouping {
container lag-interface {
if-feature lag-interface;
list lag-interface {
key "lag-interface-number";
leaf lag-interface-number {
 type uint32;
 description
   "LAG interface number";
}
uses lacp-grouping;
description
 "List of LAG interfaces";
}
description
"Container of LAG interface attributes configuration";
}
description
"Grouping for LAG interface";
}
grouping dot1q-interface-grouping {
container dot1q-interface {
leaf l2-access-type {
type identityref {
```

```
base l2-access-type;

}
description
"Encapsulation Type";
}
container dot1q {
when "'../l2-access-type'='dot1q'";
if-feature dot1q;
leaf physical-inf {
type string;
description
"Physical Interface";
}
leaf c-vlan-id {
type uint32;
description
 "VLAN identifier";
}
description
"Qot1q";
}

container qinq {
when "'../l2-access-type'='qinq'";
if-feature qinq;

leaf s-vlan-id {
type uint32;
description
 "S-VLAN Identifier";
}
leaf c-vlan-id {

type uint32;
description
 "C-VLAN Identifier";
}
description
"QinQ";
}
container qinany {
if-feature qinany;
leaf s-vlan-id {

type uint32;
description
"S-Vlan ID";
```

```
}
description
"Container for Q in Any";
}
container vxlan {
when "'../l2-access-type'='vxlan'";
if-feature vxlan;
leaf vni-id {
type uint32;
description
  "VNI Identifier";
}
leaf peer-mode {
type identityref {
base vxlan-peer-mode;
}
description
"specify the vxlan access 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 dot1Q Interface";
}
description
"Grouping for Layer2 access";
}

grouping ethernet-connection-grouping {
container connection {

leaf encapsulation-type {
type identityref {

base encapsulation-type;
}
```

```
description
"Encapsulation Type";
}
leaf-list eth-inf-type {
type identityref {

base eth-inf-type;
}
description
"Ethernet Interface Type";
}

uses dot1q-interface-grouping;
uses phy-interface-grouping;
uses lag-interface-grouping;
uses l2cp-grouping;
description
"Container for bearer";
}
description
"Grouping for bearer.";
}

grouping svc-mtu-grouping {
leaf svc-mtu {
type uint32;
description
"EVC MTU";
}
description
"Grouping for evc mtu";
}

grouping mac-addr-limit-grouping {
container mac-addr-limit {
leaf exceeding-option {
type uint32;
description
 "Exceeding option";
}
description
"Container of MAC-Addr limit configurations";
}
description
"Grouping for mac address limit";

}
```

```
grouping availability-grouping {
container availability {
leaf access-priority {
type uint32;
description
"Access priority";

}
choice redundancy-mode {
 case single-active {
   leaf single-active {
   type boolean;
   description
    "Single active";
   }
   description
   "Single active case";
 }
 case all-active {
   leaf all-active {
   type boolean;
   description
   "All active";
   }
   description
   "All active case";
 }
 description
   "Redundancy mode choice";
}
description
"Container of availability optional configurations";
}
description
"Grouping for availability";
}

grouping l2cp-grouping {
container l2cp-control {
if-feature L2CP-control;
leaf stp-rstp-mstp {
type control-mode;
description
 "STP/RSTP/MSTP protocol type applicable to all UNIs";
}
leaf pause {

type control-mode;
```

```
description
 "Pause protocol type applicable to all UNIs";

}
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
 "802.x";
}
leaf e-lmi {
type control-mode;
description
 "E-LMI";
}
leaf lldp {
type boolean;
description
 "LLDP protocol type applicable to all UNIs";
}
leaf ptp-peer-delay {
type control-mode;
description
 "PTP peer delay";
}
leaf garp-mrp {
type control-mode;
description
 "GARP/MRP";
}

leaf provider-bridge-group {
```

```
type control-mode;
description

 "Provider bridge group reserved MAC address
  01-80-C2-00-00-08";
}

leaf provider-bridge-mvrp {
type control-mode;

description
 "Provider bridge MVRP reserved MAC address
  01-80-C2-00-00-0D";
}
description
"Container of L2CP control configurations";
}
description
"Grouping for l2cp control";
}

grouping B-U-M-grouping {
container broadcast-unknown-unicast-multicast  {
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 uint32;
 description

   "the VLAN ID of the Multicast group";
}
leaf mac-gp-address {
type yang:mac-address;

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 uint32;
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 uint32;
   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 mac-loop-prevention-grouping {
container mac-loop-prevention {
leaf frequency {

type uint32;
description
 "Frequency";
}

leaf protection-type {

type identityref {
 base loop-prevention-type;
}
description
 "Protection type";
}
leaf number-retries {
type uint32;
description
 "Number of retries";
}
description
"Container of MAC loop prevention.";
}
description
"Grouping for MAC loop prevention";
}

grouping ethernet-svc-oam-grouping {
container ethernet-service-oam {
leaf md-name {
type string;
description
 "Maintenance domain name";
}
leaf md-level {
type uint8;
description
 "Maintenance domain level";
}

container cfm-802.1-ag {
list n2-uni-c {
 key "maid";
```

```
 uses cfm-802-grouping;
 description
   "List of UNI-N to UNI-C";
}
list n2-uni-n {
 key "maid";
 uses cfm-802-grouping;
 description

  "List of UNI-N to UNI-N";
}
description

 "Container of 802.1ag CFM configurations.";
}
uses y-1731;

description
"Container for Ethernet service OAM.";
}
description
"Grouping for Ethernet service OAM.";
}

grouping fate-sharing-group {
container groups {
leaf fate-sharing-group-size {
type uint16;
description
 "Fate sharing group size.";
}
list group {
key group-id;
leaf group-id {
 type string;
 description
   "Group-id the site network access
   is belonging to";
}
description
 "List of group-id";
}
description
"Groups the fate sharing group member
is belonging to";
}
description
"Grouping for Fate sharing group.";
```

```
}

grouping site-group {
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.";

}
description
"Grouping definition to assign
group-ids to site or site-network-access";
}

grouping access-diversity {
container access-diversity {
if-feature site-diversity;
uses fate-sharing-group;
container constraints {
list constraint {
 key constraint-type;
 leaf constraint-type {
   type identityref {
   base placement-diversity;
   }
   description
   "Diversity constraint type.";
 }
 container target {
   choice target-flavor {
   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.";
}
description
```

```
"This grouping defines access diversity
parameters";
}

grouping request-type-profile-grouping {
container request-type-profile {
choice request-type-choice {
case dot1q-case {
 container dot1q {
   leaf physical-if {
 type string;
 description
   "Physical interface";
   }

   leaf vlan-id {
 type uint16;
 description
   "VLAN ID";
   }
   description
  "Container for dot1q.";
 }
 description

   "Case for dot1q";
}
case physical-case {
 leaf physical-if {
 type string;
 description
  "Physical interface";
 }
 leaf circuit-id {
   type string;
   description
 "Circuit ID";
 }
 description
   "Physical case";
}
description
 "Choice for request type";
}
description
"Container for request type profile.";
}
description
```

```
"Grouping for request type profile";
}

grouping site-attachment-bearer {
container bearer {
container requested-type {
if-feature requested-type;
leaf requested-type {
 type string;
 description
   "Type of requested bearer Ethernet, DSL,
   Wireless ..Operator specific.";
}
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.
  This means no Dial access type for
  example.";
}
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.";
}
description
"Grouping to define physical properties of
```

```
a site attachment.";
}

grouping vpn-attachment-grouping {
container vpn-attachment {
leaf device-id {
type string;
description
 "Device ID";
}
container management {
leaf address-family {
 type identityref {
   base address-family;
 }
 description

   "Address family used for management.";
}
leaf address {
 type inet:ip-address;
 description
   "Management address";
}
description
 "Management configuration..";
}
choice attachment-flavor {

case vpn-flavor {
 list vpn-flavor {
 key vpn-id;
 leaf vpn-id {
   type leafref {
   path "/l2vpn-svc/vpn-services"+
   "/vpn-svc/vpn-id";
   }
   description
   "Reference to a VPN.";
 }
 leaf site-role {
   type identityref {
   base site-role;
   }
   default any-to-any-role;
   description
   "Role of the site in the IPVPN.";
 }
```

```
 description
 "List of IPVPNs attached by the Site Network Access";
}
}
case vpn-policy-id {
leaf vpn-policy-id {
type leafref {
path "/l2vpn-svc/sites/site/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.";
}
description
"Grouping for access attachment";
}

grouping site-service-basic {
container svc-input-bandwidth {
if-feature input-bw;
list input-bandwidth {
key "type";
leaf type {
 type identityref {
   base bw-type;
 }
 description
   "Bandwidth Type";
}
leaf cos-id {
type uint8;
description
"Identifier of Class of Service
 , indicated by DSCP or a CE-CLAN
 CoS(802.1p)value in the service frame.";
}
leaf vpn-id {
 type svc-id;
 description
 "Identifies the target VPN.";
```

```
}
leaf cir {
 type uint64;
 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;
 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;
 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;
 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 uint32;
 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 uint32;
 description
 "Peak Burst Size. It is measured in bytes per second.";
}
description
 "List for input bandwidth";
}
description
"From the PE perspective, the service input
bandwidth of the connection.";
```

```
}
container svc-output-bandwidth {
if-feature output-bw;
list output-bandwidth {
key "type";
leaf type {
 type identityref {
   base bw-type;
 }
 description
 "Bandwidth Type";
}
leaf cos-id {
type uint8;
description
"Identifier of Class of Service
 , indicated by DSCP or a CE-CLAN
 CoS(802.1p)value in the service frame.";
}
leaf vpn-id {
 type svc-id;
 description
 "Identifies the target VPN.";
}
leaf cir {
 type uint32;
 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 uint32;
 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 uint32;
 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 uint32;
```

```
 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 uint32;
 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 uint32;
 description
 "Peak Burst Size. It is measured in bytes per second.";
}
description
 "List for output bandwidth";
}
description
"From the PE perspective, the service output
bandwidth of the connection.";
}
description
"Grouping for site service";
}

grouping flow-definition {
container match-flow {

leaf dscp {
type inet:dscp;
description
 "DSCP value.";
}
leaf dot1p {
type uint8 {
 range "0 .. 7";
}
description
 "802.1p matching.";
}
leaf pcp {
type uint8;
description
 "PCP value";
}
```

```
leaf src-mac {
type yang:mac-address;
description

"Source MAC";
}
leaf dst-mac {
type yang:mac-address;
description
"Destination MAC";
}
container composite-id {
leaf endpoint-id {
 type string;
 description
   "Endpoint ID";
}
leaf cos-label {
 type identityref {

   base cos-id;
 }
 description
   "COS label";
}
leaf pcp {
 type uint8;
 description
   "PCP value";
}
leaf dscp {

 type inet:dscp;
 description
   "DSCP value.";
}
description
 "Container for cos color id";
}
leaf color-type {
type identityref {
 base color-type;
}
description
 "Color Types";
}
leaf-list target-sites {
type svc-id;
```

```
description
 "Identify a site as traffic destination.";
}
description
"Describe flow matching criteria.";

}
description
"Flow definition based on criteria.";
}

grouping site-service-qos-profile {
container qos {
if-feature qos;
container qos-classification-policy {
 list rule {
   key id;
   ordered-by user;
   leaf id {
   type uint16;
   description
   "ID of the rule.";

   }
   choice match-type {
   case match-flow {
   uses flow-definition;

 }
 case match-phy-port {
   leaf match-phy-port {
   type uint16;

   description
   "Defines the physical port
   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
 "Need to express marking rules ...";
}
container qos-profile {

choice qos-profile {
 description
   "Choice for QoS profile.
   Can be standard profile or custom.";
 case standard {
   leaf profile {
   type string;
   description
  "QoS Profile to be used";
   }
 }
case custom {
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 bidirection;
   description
   "The direction which QoS profile is applied to";
 }
 leaf policing {
   type identityref {
   base policing;
  }
  description
 "The policing can be either one-rate,
  two-color (1R2C) or two-rate, three-color
  (2R3C)";
```

```
 }
 leaf byte-offset {
   type uint16;
   description

 "For not including extra VLAN tags in the QoS
  calculation";

 }

 leaf rate-limit {
   type uint8;
   units percent;
   description
 "To be used if class must be rate limited.
  Expressed as percentage of the svc-bw.";
 }
 leaf discard-percentage {
   type uint8;

   default 100;
   description
 "The value of the discard-percentage,
  Expressed as pecentage of the svc-bw ";
 }
 container frame-delay {

   choice flavor {
   case lowest {

   leaf use-low-del {
   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-low-jit {
   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 fr-loss-rate {
    type decimal64 {
   fraction-digits 2;
 }
   description
   "Loss constraint on the traffic class";
   }
   description
   "Container for frame loss";
```

```
 }
 container bandwidth {
  leaf guaranteed-bw-percent {
   type uint8;
  units percent;
  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 services-grouping {
container service {
uses site-service-qos-profile;
description
"Container for service";
}
description
```

```
"Grouping for Services";
}

grouping service-grouping {
container service {
uses site-service-basic;

uses site-service-qos-profile;
description

"Container for service";
}
description
"Grouping for service.";
}

/* MAIN L2VPN SERVICE */

container l2vpn-svc {

container vpn-services {
list vpn-svc {
key "vpn-id";
leaf vpn-id {
type svc-id;
description
"Defining a service id.";
}
leaf vpn-type {
type identityref {
base service-type;
}

description
"Service type";
}
leaf customer-account-number {

type uint32;
description
"Customer account number";
}
leaf customer-name {
type string;
description
"Customer name";
}
uses svc-type-grouping;
```

```
leaf svc-topo {
type identityref {
base vpn-topology;
}
description
"Defining service topology, such as
 any-to-any,hub-spoke, etc.";
}
uses vpn-service-cloud-access;
container metro-networks {
list metro-network {

key  id;
leaf id {
 type string;
 description
   "Unique identifier for each end to end
   network connectivity in metro networks.";
}
uses inter-mkt-service;
uses intra-mkt-grouping;
description
"List of end to end network connectivity
associated with metro networks.";

}
description
"Container of Metro-Network ID configurations";
}
container global-l2cp-control {
if-feature L2CP-control;
leaf stp-rstp-mstp {
type control-mode;

description
  "STP/RSTP/MSTP protocol type applicable to all UNIs";
}
leaf pause {
type control-mode;
description
  "Pause protocol type applicable to all UNIs ";
}
leaf lldp {
type boolean;
description
  "LLDP protocol type applicable to all UNIs ";
}
description
```

```
"Container of L2CP control global configurations";
}
container service-level-mac-limit {
leaf service-level-mac-limit {
type uint16;
description
"maximum number of MAC addresses learned from
the subscriber for a single service instance
";
}
leaf action {
type identityref {
base mac-action;

}
description
"specify the action when the upper limit is
exceeded: drop the packet, flood the
packet, or simply send a warning log message.
";
}
description
"Service-level MAC-limit (E-LAN only)";
}
uses service-protection;
uses site-service;
uses vpn-service-multicast;
uses vpn-extranet;
description
"List of vpn-svc";
}
description
"Container for VPN services.";
}

/* SITE */
container sites {
list site {
key "site-id site-type";
leaf site-id {
type string;
description
"Site id";
}
leaf site-type {
type identityref {
base site-type;
}
```

```
description
"Site type";
}
uses site-device;
uses customer-location-info;
uses site-management;
uses site-diversity;
uses site-vpn-policy;
container signaling-options {
if-feature signaling-options;
uses signaling-options-grouping;
description
"Container for signaling option";

}
container load-balance-options {
uses load-balance-grouping;

description
"Container for load balance options";
}
uses services-grouping;
uses B-U-M-grouping;
uses site-security;
uses operational-requirements-ops;
container site-network-accesses {
list site-network-accesse {
key "network-access-id";
leaf network-access-id {
  type string;
  description
  "Identifier of network access";
}

leaf remote-carrier-name {
  when "'../site-type' = 'enni'" {
   description
   "Site type = enni";
  }
  type string;
  description
  "Remote carrier name";
}
uses access-diversity;
uses site-attachment-bearer;
uses ethernet-connection-grouping;
uses svc-mtu-grouping;
uses availability-grouping;
```

```
uses vpn-attachment-grouping;
uses service-grouping;
uses B-U-M-grouping;
uses ethernet-svc-oam-grouping;
uses site-security;
description
  "List of Site Network Accesses.";
}
description
"Container of Site Network Access configurations";
}
description
"List of sites";
}

description
"Container of site configurations";
}

description

"Container for L2VPN service";
}
}
```

<CODE ENDS>

9. Security Considerations

The YANG modules defined in this document MAY be accessed via the RESTCONF protocol [RFC8040] or NETCONF protocol ([RFC6241]). The lowest RESTCONF or NETCONF layer requires that the transport-layer protocol provides both data integrity and confidentiality, see Section 2 in [RFC8040] and [RFC6241]. The client MUST carefully examine the certificate presented by the server to determine if it meets the client's expectations, and the server MUST authenticate client access to any protected resource. The client identity derived from the authentication mechanism used is subject to the NETCONF Access Control Module (NACM) ([RFC6536]). Other protocols to access this YANG module are also required to support the similar mechanism.

The data nodes defined in the "ietf-l2vpn-svc" YANG module MUST be carefully created/read/updated/deleted. The entries in the lists below include customer proprietary or confidential information, therefore only authorized clients MUST access the information and the other clients MUST NOT be able to access to the information.

- /l2vpn-svc/customer-info/customer-info

- /l2vpn-svc/vpn-services/vpn-svc
- /l2vpn-svc/sites/site

10. 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.

This document has drawn on the work of the L3SM Working Group expressed in [RFC8049].

11. 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: L2SM WG
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
```

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>.

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

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

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

[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>.

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

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7224] Bjorklund, M., "IANA Interface Type YANG Module", RFC 7224, DOI 10.17487/RFC7224, May 2014, <https://www.rfc-editor.org/info/rfc7224>.

[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>.

[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>.

[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>.

[RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <https://www.rfc-editor.org/info/rfc8049>.

12.2. Informative References

[I-D.ietf-bess-evpn-yang]
Brissette, P., Sajassi, A., Shah, H., Li, Z., Tiruveedhula, K., Hussain, I., and J. Rabadan, "Yang Data Model for EVPN", draft-ietf-bess-evpn-yang-02 (work in progress), March 2017.

[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-06 (work in progress), June 2017.

[I-D.ietf-opsawg-service-model-explained]
Wu, Q., LIU, W., and A. Farrel, "Service Models Explained", draft-ietf-opsawg-service-model-explained-03 (work in progress), September 2017.

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

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

[MEF-23-2] MEF Forum, "Implementation Agreement MEF 23.2 : Carrier Ethernet Class of Service - Phase 3", August 2016.

[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>.

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.

Appendix B. Open Issues

- Do we need to support L2VPN Interworking with ATMoMPLS,PPPoMPLS, FroMPLS?
- Need to understand relationship between member link name under LAG-interface and network-access-id.

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