L2SM Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 19, 2017

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

February 15, 2017

A YANG Data Model for L2VPN Service Delivery
draft-wen-l2sm-l2vpn-service-model-04

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 use 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 http://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 August 19, 2017.

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 use 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 modelled 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.wu-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 modelling architecture.

1.1. Terminology

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

o 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.

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

UNI: User 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].

3.1. Applicability of the Layer 2 VPN Service Model

The L2SM defined in this document applies to both point-to-point (E-Line) and multipoint-to-multipoint (E-LAN) carrier Ethernet services.

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.

Rather than introducing a new set of terminologies, the L2SM will align with existing MEF attributes when it’s applicable. Therefore, service providers can easily integrate any new application that leverages the L2SM data (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

A Layer 2 VPN circuit can be port-based, in which case any service frames received from a subscriber within the contracted bandwidth will be delivered to the corresponding remote site, regardless of the customer VLAN value (C-tag) of the incoming frame. 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 service-multiplexing attribute with the connection type (point-to-point or multipoint-to-multipoint), a Layer 2 VPN 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 All-to-One Bundling. All unicast/broadcast/multicast packets are delivered unconditionally over the EVC. No service multiplexing is allowed on an EPL UNI.

  EVPL: On the other hand, an 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 Service Network Topology

Figure 1depicts a typical service provider’s physical network topology. Most service providers have deployed an IP, MPLS, or Segment Routing (SR) multi-service core infrastructure. Customer Edge (CE) devices are placed at customer premises as demarcation points that backhaul in-profile service frames from the subscriber over the access network to the Provider Edge (PE) equipment. The actual transport technology or physical topology between CE and PE is outside the scope of the L2SM model.

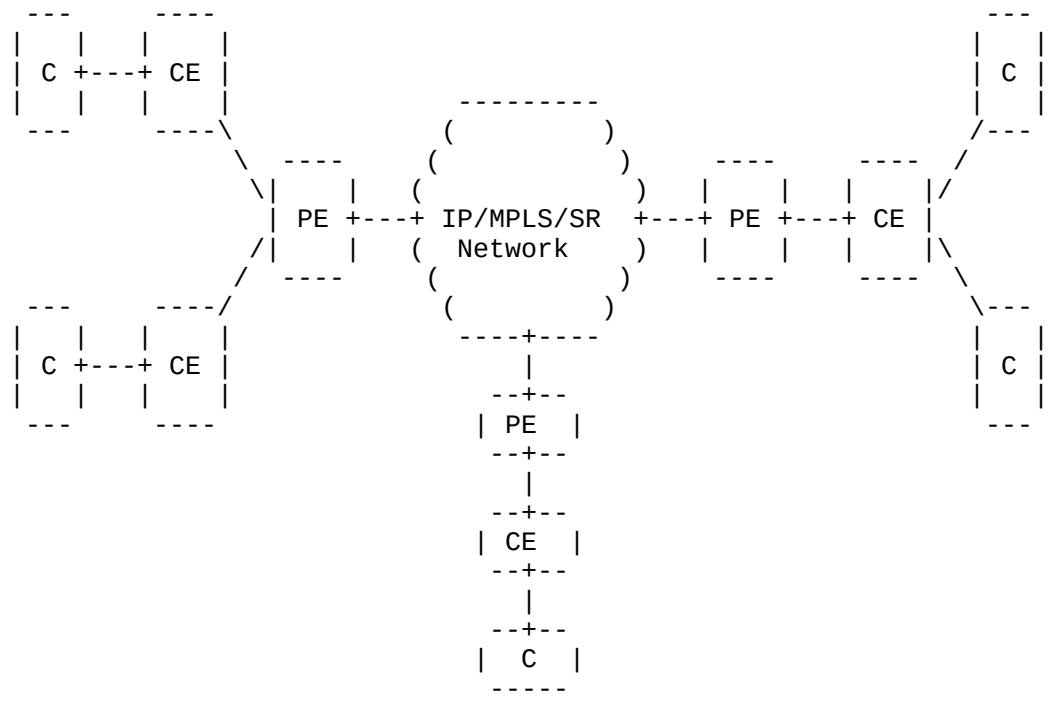

Figure 1: Reference Network for the Use of the L2VPN Service Model

From the subscriber perspective, however, all the edge networks 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.

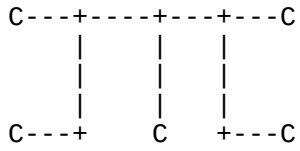

Figure 2: Customer View of the L2VPN

3.4. Layer 2 VPN Ethernet Virtual Circuit Construct

The base model of EVC is shown in Figure 3.

Subscriber edge network device (C) connects to the service provider's CE equipment. The link between C and CE devices is referred as the User Network Interface (UNI). For clarification, this is called the UNI-C on the subscriber side and UNI-N on the provider side.

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

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

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

Typically, the CE device at customer premises is a layer 2 Ethernet switch or NID. 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 CE 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.

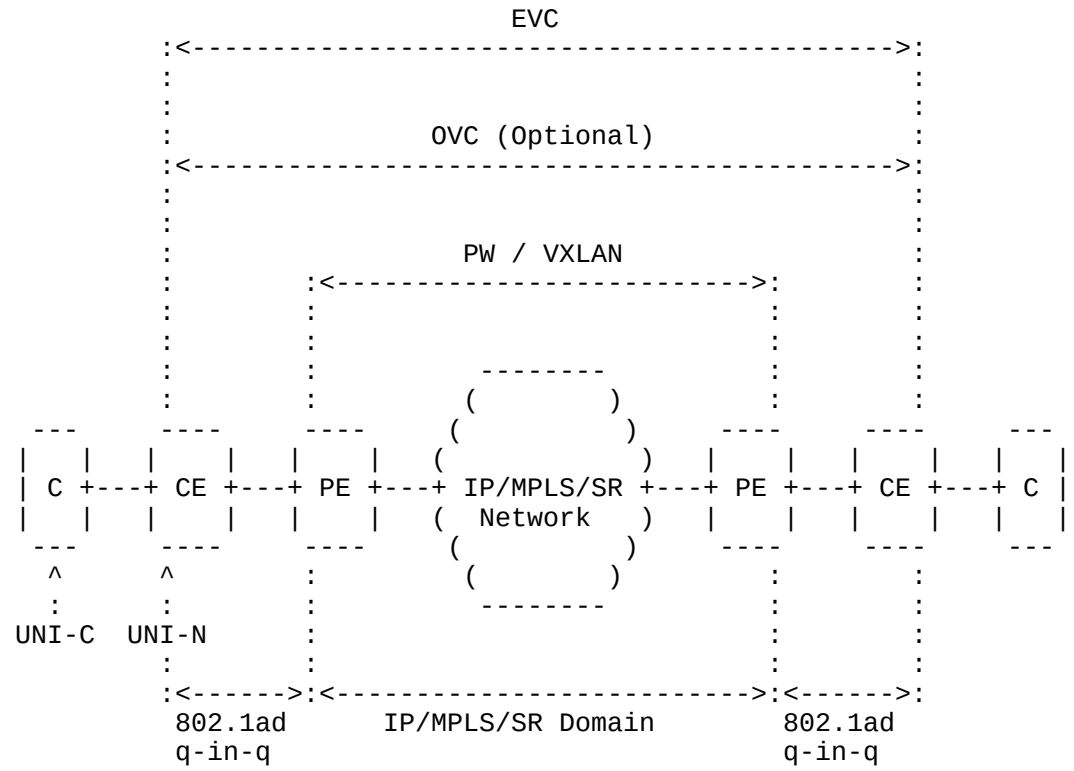

Figure 3: Architectural Model for EVC over a Single Network

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 the 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).

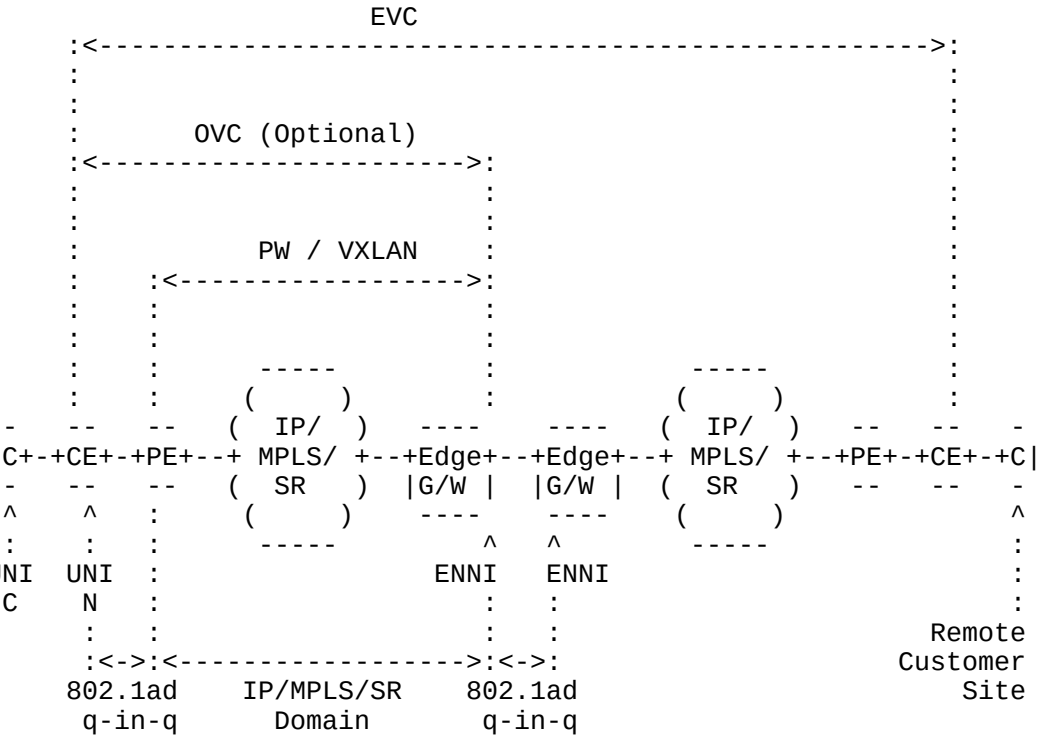

Figure 4: Architectural Model for EVC over Multiple Networks

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.wu-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 [I-D.ietf-l3sm-l3vpn-service-model].

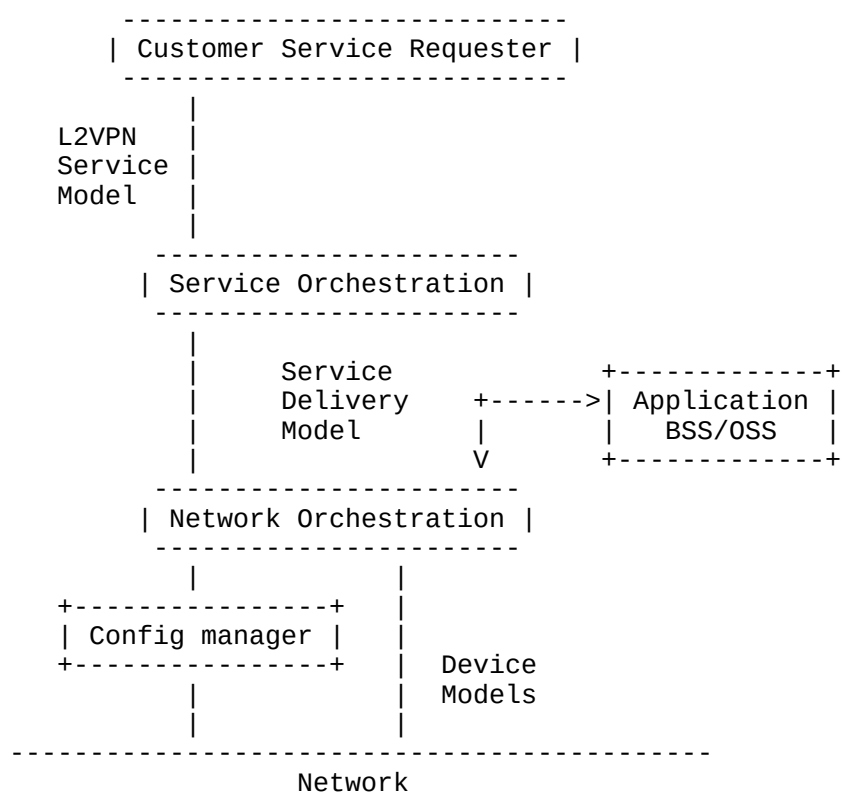

Figure 5: Reference Architecture for the Use of the L2VPN Service Model

Additionally, this data model can be compared with the service delivery models described in [I-D.ietf-bess-l2vpn-yang] and [I-D.ietf-bess-evpn-yang] as discussed in Section 6.

5. Design of the Data Model

The YANG module is divided in three main containers: customer-info, vpn-services, and sites.

The customer-info defines global parameters for a specific customer.

The vpn-svc container under vpn-services defines global parameters for the VPN service for a specific customer.

A site contains at least one port (i.e., ports providing access to the sites defined in Section 5.1.3.1.8) and there may be multiple

ports in case of multihoming. The site to port 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.

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

```
module: ietf-l2vpn-svc

+--rw l2vpn-svc
   +--rw customer-info
   |  +--rw customer-info* [customer-account-number customer-name]
   |     +--rw customer-account-number          uint32
   |     +--rw customer-name                    string
   |     +--rw customer-operation-center
   |        +--rw customer-noc-street-address?  string
   |        +--rw customer-noc-phone-number
   |           +--rw main-phone-num?            uint32
   |           +--rw extension-options?         uint32
   +--rw vpn-services
   |  +--rw vpn-svc* [vpn-id]
   |     +--rw vpn-id                           svc-id
   |     +--rw svc-type?                        identityref
   |     +--rw evc-type
   |     |  +--rw evc-id?                       svc-id
   |     |  +--ro number-of-pe?                 uint32
   |     |  +--ro number-of-site?               uint32
   |     |  +--rw uni-list {uni-list}?
   |     |     +--rw uni-list* [network-access-id]
   |     |        +--rw network-access-id       string
   |     +--rw ovc-type {ovc-type}?
   |     |  +--rw ovc-list* [ovc-id]
   |     |     +--rw ovc-id                     svc-id
   |     |     +--rw on-net?                    boolean
   |     |     +--rw off-net?                   boolean
   |     +--rw ethernet-svc-type
   |     |  +--rw (ethernet-svc-type)?
   |     |     +--:(e-line)
   |     |     |  +--rw epl?                    boolean
   |     |     |  +--rw evpl?                   boolean
   |     |     +--:(e-lan)
   |     |     |  +--rw ep-lan?                 boolean
```

```
|     |     |  +--rw evp-lan?                boolean
|     |     +--:(e-access)
|     |        +--rw access-epl?             boolean
|     |        +--rw access-evpl?            boolean
|     +--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 ce-vlan-preservation
|     +--rw metro-network-id
|     |  +--rw inter-mkt-service?            boolean
|     |  +--rw intra-mkt* [metro-mkt-id mkt-name]
|     |     +--rw metro-mkt-id               uint32
|     |     +--rw mkt-name                   string
|     |     +--rw ovc-id?                    string
|     +--rw L2CP-control
|     |  +--rw stp-rstp-mstp?                control-mode
|     |  +--rw pause?                        control-mode
|     |  +--rw lldp?                         boolean
|     +--rw load-balance-options
|     |  +--rw fat-pw?                       boolean
|     |  +--rw entropy-label?                boolean
|     |  +--rw vxlan-source-port?            string
|     +--rw svlan-id-ethernet-tag?           string
|     +--rw cvlan-id-to-evc-map* [evc-id type]
|     |  +--rw evc-id       -> /l2vpn-svc/vpn-services/vpn-svc/vpn-id
|     |  +--rw type                          identityref
|     |  +--rw cvlan-id* [vid]
|     |     +--rw vid                        identityref
|     +--rw service-level-mac-limit?         string
|     +--rw service-protection
|     |  +--rw protection-model
|     |  +--rw peer-evc-id
|     +--rw sla-targets
|
+--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 site-name?                 string
         |     +--rw management
         |        +--rw address?                inet:ip-address
         |        +--rw management-transport?   identityref
         +--rw managemnt
         |  +--rw type?                         identityref
         +--rw location
         |  +--rw address?                      string
         |  +--rw zip-code?                     string
         |  +--rw state?                        string
         |  +--rw city?                         string
         |  +--rw country-code?                 string
         +--rw site-diversity {site-diversity}?
         |  +--rw groups
         |     +--rw group* [group-id]
         |        +--rw group-id                string
         +--rw vpn-policies
         |  +--rw vpn-policy* [vpn-policy-id]
         |     +--rw vpn-policy-id              string
         |     +--rw entries* [id]
         |        +--rw id                      string
         |        +--rw filter
         |        |  +--rw (lan)?
         |        |     +--:(lan-tag)
         |        |        +--rw lan-tag*       string
         |        +--rw vpn
         |           +--rw vpn-id
         |           |        -> /l2vpn-svc/vpn-services/vpn-svc/vpn-id
         |           +--rw site-role?           identityref
         +--rw signaling-option {signaling-option}?
         |  +--rw signaling-option* [type]
         |     +--rw type                       identityref
         |     +--rw mp-bgp-l2vpn
         |     |  +--rw vpn-id?                 svc-id
         |     |  +--rw type?                   identityref
         |     +--rw mp-bgp-evpn
         |     |  +--rw vpn-id?                 svc-id
         |     |  +--rw type?                   identityref
         |     |  +--rw mac-learning-mode?      identityref
         |     |  +--rw arp-suppress?           boolean
         |     +--rw t-ldp-pwe
         |     |  +--rw PE-EG-list* [service-ip-lo-addr vc-id]
```

```
|     |     +--rw service-ip-lo-addr   inet:ip-address
|     |     +--rw vc-id                string
|     +--rw pwe-encapsulation-type
|     |  +--rw ethernet?               boolean
|     |  +--rw vlan?                   boolean
|     +--rw pwe-mtu
|     |  +--rw allow-mtu-mismatch?     boolean
|     +--rw control-word
+--rw load-balance-options
|  +--rw fat-pw?                       boolean
|  +--rw entropy-label?                boolean
|  +--rw vxlan-source-port?            string
+--ro actual-site-start?               yang:date-and-time
+--ro actual-site-stop?                yang:date-and-time
+--rw ports
   +--rw port* [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 request-type-profile
      |  |     +--rw (request-type-choice)?
      |  |        +--:(dot1q-case)
      |  |        |  +--rw dot1q
      |  |        |     +--rw physical-if?   string
      |  |        |     +--rw vlan-id?       uint16
      |  |        +--:(physical-case)
      |  |           +--rw physical-if?      string
      |  |           +--rw circuit-id?       string
```

```
|  +--rw always-on?          boolean {always-on}?
|  +--rw bearer-reference?   string {bearer-reference}?
+--rw ethernet-connection
|  +--rw ESI?                      string
|  +--rw interface-description?  string
|  +--rw vlan
|  |  +--rw vlan-id?               uint32
|  +--rw dot1q
|  |  +--rw physical-inf?          string
|  |  +--rw vlan-id?               uint32
|  +--rw qinq
|  |  +--rw s-vlan-id?             uint32
|  |  +--rw c-vlan-id?             uint32
|  +--rw sub-if-id?                uint32
|  +--rw vxlan
|  |  +--rw vni-id?                uint32
|  |  +--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 encapsulation-type?    enumeration
|  |  +--rw ethertype?             string
|  |  +--rw lldp?                  boolean
|  |  +--rw oam-802.3AH-link {oam-3ah}?
|  |  |  +--rw enable?             boolean
|  |  +--rw uni-loop-prevention?   boolean
|  +--rw 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 encapsulation-type?  enumeration
|           +--rw ethertype?           string
|           +--rw lldp?                boolean
+--rw 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 evc-mtu?                   uint32
+--rw availability
|  +--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-id)
|        +--rw vpn-id?
|        |      -> /l2vpn-svc/vpn-services/vpn-svc/vpn-id
|        +--rw site-role?        identityref
+--rw service
|  +--rw svc-input-bandwidth {input-bw}?
|  |  +--rw input-bandwidth* [id type]
|  |     +--rw id                string
```

```
|  |     +--rw type              identityref
|  |     +--rw evc-id?           svc-id
|  |     +--rw CIR?              uint32
|  |     +--rw CBS?              uint32
|  |     +--rw EIR?              uint32
|  |     +--rw EBS?              uint32
|  |     +--rw CM?               uint32
|  +--rw svc-output-bandwidth {output-bw}?
|  |  +--rw output-bandwidth* [id type]
|  |     +--rw id                string
|  |     +--rw type              identityref
|  |     +--rw evc-id?           svc-id
|  |     +--rw CIR?              uint32
|  |     +--rw CBS?              uint32
|  |     +--rw EIR?              uint32
|  |     +--rw EBS?              uint32
|  |     +--rw CM?               uint32
|  +--rw svlan-id-ethernet-tag?  string
|  +--rw cvlan-id-to-evc-map* [evc-id type]
|  |  +--rw evc-id
|  |  |         -> /l2vpn-svc/vpn-services/vpn-svc/vpn-id
|  |  +--rw type                 identityref
|  |  +--rw cvlan-id* [vid]
|  |     +--rw vid               identityref
|  +--rw service-level-mac-limit?          string
|  +--rw service-multiplexing?   boolean
|  +--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 cos-color-id
|     |     |  |     |  +--rw device-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 ingress-profile?    string
|           |  +--rw egress-profile?     string
|           +--:(custom)
|              +--rw classes {qos-custom}?
|                 +--rw class* [class-id]
|                    +--rw class-id      string
|                    +--rw direction?    direction-type
|                    +--rw policing?     identityref
|                    +--rw byte-offset?  uint16
|                    +--rw perf-tier-opt? identityref
|                    +--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 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-filtering
   +--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
   +--rw B-U-M-storm-control
      +--rw BUM-overall-rate?           uint32
      +--rw BUM-rate-per-type* [type]
         +--rw type                     identityref
         +--rw rate?                    uint32
```

Figure 6

5.1. Overview of Main Components of the Model

The L2SM model is structured in a way that allows the provider to list multiple circuits of various service types for the same subscriber.

5.1.1. Customer Information

The "customer-info" container contains 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.

The subscriber operation center and main contact number are also listed here for reference purpose.

5.1.2. 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:

Service Type (vpn-type): Used to indicate service Type. The identifier is a string allowing to any encoding for the local administration of the VPN service.

Ethernet Connection Service Type (eth-svc-type): used to identify supported Ethernet Connection Service Types.

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.

VPN Signaling (vpn-signaling-option): Defines which protocol or signaling must be activated between the subscriber and the provider.

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

SVLAN ID Ethernet Tag: Used to identify the service-wide "normalized S-tag".

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.

5.1.2.1. Service Type

The "svc-type" supports two service types: one for EVC (Ethernet Virtual Connection) and the other for OVC (Operator Virtual Connection). These two parameters are not mutually exclusive. Depending on the service-type, a Layer 2 VPN service may be identified by EVCtype, OVCtype, or both. E-Line and E-LAN providers shall have an EVC-ID assigned to the UNI-to-UNI circuit. If the service has remote UNIs in an off-net partner’s network, there will be one OVC-ID for the on-net segment between the local UNI and the E-NNI interconnect, and one OVC-ID for each off-net segment from E-NNI to the remote UNI.

E-Access, on the other hand, is an OVC-based service. The E-Access service provider will assign an OVC-ID for the circuit between UNI and E-NNI.

New service types could be added by augmentation.

5.1.2.1.1. EVC

The "evc" case contains an "evc-id" leaf and "uni-list" container. And the "vpn-id" will be associated with the "evc-id". Only one "evc-id" is allowed for each "vpn-id". The "evc-id" leaf will be specified for E-Line and E-LAN service types. "uni-list" will specify the UNI list associated with the same EVC service.

The EVC-ID is intended to be a structured string. Each service provider can decide the nomenclature in its network. In addition, "number of PEs" and "number of sites" can be specified under the "evc" container.

5.1.2.1.2. OVC

The "ovc" case contains "ovc-list". For each "ovc-list" entry, there are two boolean subcases ("on-net" and "off-net") and one "ovc-id" leaf is specified.

For E-Access or services with off-net UNIs, the "on-net" leaf MUST be marked TRUE, and the "ovc-id" will be specified.

In case of E-Access, the "vpn-id" will be associated with the on-net "ovc-id". Only one on-net "ovc-id" is allowed for each "vpn-id".

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. However, the "vpn-id" is still associated with the "evc-id". Only one "evc-id" is allowed for each "vpn-id".

5.1.2.2. Ethernet Connection Service Type

The "ethernet-svc-type" group contains all supported Ethernet connection service types. One, and only one, "ethernet-svc-type" is selected for each "vpn-id".

The currently supported Ethernet Connection service types are listed in Section 3.2. New Ethernet Connection service types can be added in the future.

5.1.2.3. 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), and hub and spoke disjoint (where hubs cannot exchange traffic). New topologies could be added by augmentation. By default, the any-to-any VPN service topology is used.

5.1.2.4. 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 5.1.3 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.

5.1.2.5. Metro Network Partition

Some service providers may divide their networks into multiple administrative domains. And a Layer 2 VPN service may span across more than one metro network belonging to the same service provider. The optional "metro-network-id" 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.

5.1.2.6. 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. Along with the great revenue opportunities, 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 Nx10G or Nx100G 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 considered a single 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".

5.1.2.7. SVLAN ID Ethernet Tag

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

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.)

The purpose of the optional "svlan-id-ethernet-tag" leaf is to identify the service-wide "normalized S-tag".

5.1.2.8. CVLAN ID To EVC MAP

When more than one service is multiplexed onto the same interface, ingress service frames are conditionally transmitted through one of the EVC/OVCs based upon pre-arranged customer VLAN to EVC mapping. Multiple customer VLANs can be bundled across the same EVC.

"cvlan-id-to-evc-map", when applicable, 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 tag (the C-tag).

5.1.2.9. 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 subscriber 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 subscriber MAC address limit and information to describe the action when the limit is exceeded.

5.1.2.10. Service Protection

Sometimes the subscriber 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 EVC or OVC will be protected by a backup EVC 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.

- 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.

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

An "peer-evc-id" should be specified when the "protection-model" has been set.

5.1.3. site

The "site" container is used for the provider to store information of detailed implementation arrangements made with either the subscriber 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.

There are two possible types of external facing connections associated with an Ethernet VPN service. These give rise to two different types of site at which the connection is made:

- UNI site: where a customer edge device connects to one or more VPN services.

- E-NNI site: where two Ethernet service providers inter-connect with each other.

Most of the attributes of a site are common to the two typs of site and so are presented just once. Divergences (that is, attributes that are specific to the type of site) are captured in type-dependent containers. In the text that follows, the phrase "between the subscriber and the provider" is used to follow the more common case of a UNI site, but should also be taken to apply to "between two providers" in the E-NNI case.

For each site, there are sub-containers to maintain physical link attributes, service frame and Layer 2 control protocol frame disposition, Ethernet service OAM attributes, and agreements for service bandwidth profiles and priority levels.

5.1.3.1. Generic Site Objects

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.

Site signaling (signaling-options): Defines which protocol or signaling must be activated between the subscriber and the provider.

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

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

5.1.3.1.1. Site ID

The "site-id" leaf contains an arbitrary string to uniquely identify the site within the overall network infrastructure. The format of the site-id is determined by the local administration of the VPN service.

5.1.3.1.2. 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.

5.1.3.1.3. Site Location

The information in the "location" sub-container under a "site" 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. Location is expressed in terms of postal information.

5.1.3.1.4. Site Diversity

Some subscribers may request upstream PE diversity between two or more sites. These sites will share the same diversity group ID under the optional "site-diversity" sub-container.

5.1.3.1.5. Site Security

This sub-container presents parameters for ingress service stream admission control and encryption profile information. It is also a placeholder for further site-security options that may be added by augmentation.

5.1.3.1.6. Site 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 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.

The "signaling-option" list uses the combination of "name" and "type" as the key. The "name" leaf is a free-form string of the VPN instance name. The "type" leaf is for the signaling protocol: BGP-L2VPN, BGP-EVPN, or T-LDP.

5.1.3.1.6.1. 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.

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, only the target community (or communities) are listed at the service level.

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

5.1.3.1.6.2. 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, only the target community (or communities) are listed at the service level.

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

5.1.3.1.6.3. 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".

5.1.3.1.6.4. 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" container contains two Boolean type leaves: "ethernet" and "ethernet-vlan". If "signaling-option" is "mp-bgp-l2vpn" or "t-ldp-pwe", then exactly one of "ethernet" and "ethernet-vlan" MUST be marked TRUE .

5.1.3.1.6.5. 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.

5.1.3.1.6.6. 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.

5.1.3.1.7. Site Load Balance Options

See Section 5.1.2.6.

5.1.3.1.8. Ports

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

5.1.3.1.8.1. ID

"id" is a free-form string to identify a given interface. The service provider can decide on the actual nomenclature used in the management systems.

5.1.3.1.8.2. Remote Carrier Name

Remote Carrier Name is the Name of Remote Carrier associated with the remote end of an E-NNI and so only applies for that type of VPN connectivity.

5.1.3.1.8.3. Access Diversity

In order to help the different placement scenarios, a site-network-access (i.e., port defined in Section 5.1.3.1.8) may be tagged using one or more fate sharing group identifiers. The fate sharing group identifier is a string so it can accommodate both explicit naming of a group of sites (e.g. "multihomed-set1") or a numbered identifier (e.g. 12345678). The meaning of each group-id is local to each customer administrator.

5.1.3.1.8.4. 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.

5.1.3.1.8.5. Ethernet Connection

The ethernet-connection container presents two sets of link attributes: physical or optional LAG interface attributes. These parameters are essential for the connection between subscriber and provider edge devices to establish properly.

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. "encapsulation-type" is for user to select between Ethernet encapsulation (port-based) or Ethernet VLAN encapsulation (VLAN-based). All allowed Ethertypes of ingress service frames can be listed under "ethertype". In addition, the subscriber 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.

Sometimes, the subscriber 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 subscriber 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 subscriber 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. Subscriber 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.

If the Ethernet service is enabled on a logical unit on the connection at the interface, the "sub-if-id" should be specified.

The "Ethernet-connection" container also presents site specific (S-tag, C-tag) management options. The overall S-tag for the Ethernet circuit and C-tag to EVC mapping, if applicable, has been placed in the service container. The S-tag under "port" 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 EVC has been previously assigned to another service thus can not be used by this attachment circuit. Another use case is when multiple E-access OVCs from the same E-NNIs are attached to the same E-LAN service.

The "svlan-id-ethernet-tag" in the "Ethernet-connection" container is either the S-tag inserted at a UNI or the outer tag of ingress

packets at an E-NNI. These parameters are included in the L2SM to facilitate other management system to generate proper configuration for the network elements.

The "ethernet-connection" container also contains an optional site-specific C-tag to EVC mapping.

5.1.3.1.8.6. EVC MTU

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

5.1.3.1.8.7. 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.

5.1.3.1.8.8. Availability

EVPN supports PE geo-redundancy in the access domain. The connection between a multi-homed CE to PE is identified with a uniquely assigned ID referred as an Ethernet Segment Identifier (ESI). Because a learned MAC address is propagated via BGP, it allows for multiple active paths in forwarding state and for load-balancing options.

The "availability" container contains ESI and redundancy mode attributes for an EVPN multi-homing site.

5.1.3.1.8.9. L2CP-Control

To facilitate interoperability between different Multiple System Operators (MSOs), the MEF has provided normative guidance on Layer 2 Control Protocol (L2CP) processing requirements for each service type. Subscriber and provider should make pre- arrangement on whether to allow interaction between the edge device or keep each other’s control plane separate on a per-protocol basis.

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.

5.1.3.1.8.10. Service

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

5.1.3.1.8.10.1. Bandwidth

The service bandwidth refers to the bandwidth requirement between PE and CE. 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 port 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 bandwidth container may also include a "cos-id" parameter. If the "cos-id" is not present within the bandwidth container, the bandwidth is per evc, which provides rate enforcement for all ingress service frames at the interface that are associated with a particular EVC.

If the "cos-id" is present, the bandwidth is per CoS, which provides rate enforcement for all service frames for a given class of service. The class of service is identified via a CoS identifier. So this bandwidth profile applies to service frames over an EVC with a

particular CoS value. Multiple input/output-bandwidthper-cos-id can be associated with the same EVC.

5.1.3.1.8.10.2. 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.

5.1.3.1.8.10.2.1. QoS Classification

In MEF 23.2 ([MEF-23-2]) three types of model are defined as the following:

Class-of-Service Identifier based on EVC or OVC EP (End Point): In this model, regardless of customer marking, all in-profile frames will be marked with the service level in the contractual agreement. Customer CoS markings are preserved throughout the provider network. The bandwidth profile consists of one set of CIR/CBS and EIR/EBS values.

Class-of-Service Identifier based on Priority Code Point: Using this model, multiple classes of services can be associated with a single customer EVC, identified by dot1p bits in the C-tag. Each service level has its own individual bandwidth profile. Out-of-profile packets will be discarded. Customer CoS markings are preserved.

Class-of-Service Identifier based on DSCP: Using this model, multiple classes of service can be associated with a single customer EVC, identified by DSCP bits in the IP header. Each service level has its own individual bandwidth profile. Out-of-profile packets will be discarded. Customer CoS markings are preserved.

Similarly, the cos-color-id can be assigned based on EVC or OVC EP, dot1p value in C-tag, or DSCP in IP header. Ingress service frames are metered against the bandwidth profile based on the cos-identifier. A "color" will be assigned to a service frame to identify its bandwidth 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.

Ingress/egress-bandwidth-profile-per-evc presents the ingress/egress bandwidth profile per EVC, providing rate enforcement for all ingress service frames at the interface that are associated with a particular EVC.

Alternately, ingress/egress-bandwidth-profile-per-cos-id presents the ingress/egress bandwidth profile per CoS, providing rate enforcement for all service frames for a given class of service. The class of service is identified via a CoS identifier. So this bandwidth profile applies to service frames over an EVC with a particular CoS value. Multiple ingress/egress-bandwidth-profile-per-cos-id can be associated with the same EVC.

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.). 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.

5.1.3.1.8.10.2.2. 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:

- 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.

- o 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.

- o 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.

5.1.3.1.8.11. Security Filtering

5.1.3.1.8.11.1. BUM Strom Control

For point-to-point E-LINE 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 in-profile 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.

5.1.3.1.8.11.2. MAC Loop Protection

MAC address flapping between different physical ports typically indicates a bridge loop condition in the subscriber 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.

5.1.3.1.8.11.3. Service Level MAC Limit

See Section 5.1.2.10.

5.1.3.1.8.12. 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 carrier network, particularly in the area of service availability and Mean Time To Repair (MTTR). Ethernet Service OAM in the L2SM 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 EVC 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 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 subscriber’s network, solely within the provider’s network, interact between the subscriber-to-provider or provider-to-provider edge equipment, or tunnel over another provider’s 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 "port" 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 "port" 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.

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.wu-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:

- VRF 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.

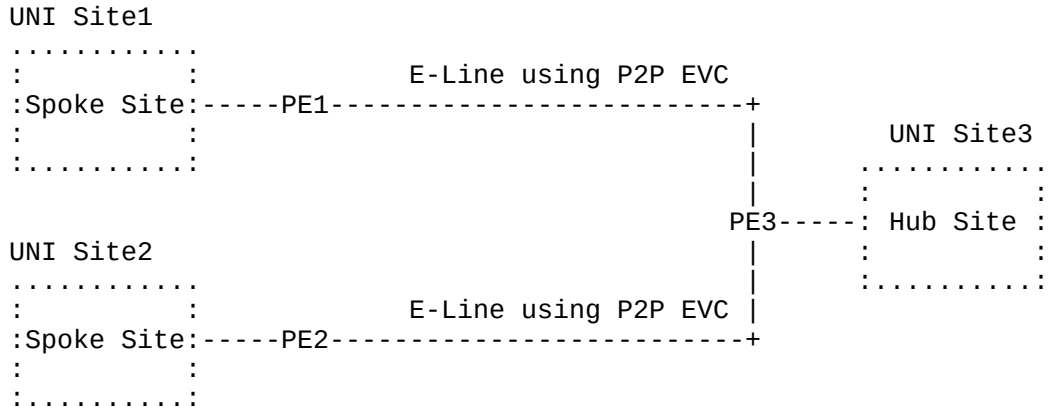

Figure 7: Reference Network for Simple Example

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

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

<vpn-service>
  <vpn-id>12456488</vpn-id>
  <svc-id-type>EVC</svc-id-type >
  <evc>
    <uni-list>
      <uni>UNI2</uni>
      <uni>UNI3</uni>
    </uni-list>
  </evc>
  <eth-svc-type>eline</eth-svc-type>
  <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 Taregts (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>VRF</type>
      <mp-bgp-l2vpn>
         <svc-id>12456487</svc-id>
         <type>VPWS</type>
      </mp-bgp-l2vpn>
    </signaling-option>
  </signaling-options>
  <site-ports>
    <site-port>
      <site-port-id>Spoke_UNI-Site1</site-port-id>
      <access-diversity>
        <groups>
          <group>
            <group-id>20</group-id>
          </group>
        </groups>
      <access-diversity>
      <ethernet-connection>
        <vlan>
          <vlan-id>17</vlan-id>
        </vlan>
        <physical-interface>
          <encapsulation-type>ETH</encapsulation-type>
        </physical-interface>
      </ethernet-connection>
      <service>
        <svc-ingress-bandwidth>
          <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-port>
                </site-ports>
                <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 5.1.3.1.3, 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 VRF is already present that fits the needs. If not, it will provision the VRF: 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:

```
ip vrf Customer1
  export-map STD-CUSTOMER-EXPORT      <---- Standard SP configuration
  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
!
```

When the VRF has been provisioned, the management system can start configuring the access on the PE using the allocated interface information. The VLAN tag is chosen by the management system. One VLAN 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:

```
interface GigabitEthernet0/0/0/3.100 l2transport
  encapsulation dot1ad 100
  rewrite ingress tag pop 1 symmetric
  l2vpn
  xconnect group EPL
  p2p EPL
  interface GigabitEthernet0/0/0/3.100
  neighbor 100.100.100.1 pw-id 100
!
interface GigabitEthernet4/8
  no ip address
  speed nonegotiate
  no keepalive
  ethernet dot1ad nni
  service instance 100 ethernet
  encapsulation dot1q 100
  rewrite ingress tag pop 1 symmetric
  xconnect 100.100.100.2 100 encapsulation mpls
```

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 GigabitEthernet0/9
  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@2017-02-10.yang"
module ietf-l2vpn-svc {
  namespace "urn:ietf:params:xml:ns:yang:ietf-l2vpn-svc";
  prefix "l2svc";

  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: Bin_Wen@comcast.com";
     description
       "The YANG module defines a generic service configuration
        model for Layer 2 VPN services common across all of the
        vendor implementations.";
     revision 2017-02-13{
       description
         "Initial revision -04 version";
       reference
         "draft-wen-l2sm-l2vpn-service-model-04.txt
          A YANG Data Model for L2VPN Service Delivery.";
     }

     /* Features */

     feature input-bw {
       description
         "Input Bandwidth";
     }
     feature output-bw {
       description
         "Output Bandwidth";
     }
     feature uni-list {
       description
         "Enable support UNI list";
     }
     feature ovc-type {
       description
         "Enable support OVC type";
     }
     feature cloud-access {
       description
         "Allow VPN to connect to a Cloud Service
          provider.";
     }
     feature oam-3ah {
       description
         "Enables support of OAM 802.3ah";
     }
     feature Micro-BFD {
       description
         "Enables support of Micro-BFD";
     }
     feature bfd {
       description
```

```
        "Enables support of BFD";
    }
    feature signaling-option {
      description
        "Enable support of signaling option";
    }
    feature site-diversity {
      description
        "Enables support of site diversity constraints";
    }
    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";
    }

    /* Typedefs */

    typedef svc-id {
      type string;
      description
        "Service ID";
    }
    typedef direction-type {
      type string;
      description
```

```
        "Direction";
    }
    typedef evc-id-type {
      type string;
      description
        "EVC ID type";
    }
    typedef ovc-id-type {
      type string;
      description
        "OVC ID type";
    }
    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";
    }
    typedef neg-mode {
      type enumeration {
        enum full-duplex {
          description
            "Full duplex mode";
        }
        enum auto-neg {
          description
            "Auto negotiation mode";
        }
```

```
      }
      description
        "Defining a type of the negotiation mode";
    }

    /* Identities */

    identity bw-type {
      description
        "Identity of bandwidth";
    }
    identity bw-per-cos {
      base bw-type;
      description
        "Bandwidth is per cos";
    }
    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 evc {
      base service-type;
      description
        "EVC type.";
    }
    identity ovc {
      base service-type;
      description
        "OVC 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 vrf {
      base vpn-signaling-type;
      description
```

```
        "Virtual routing and forwarding (VRF).";
    }
    identity vfi {
      base vpn-signaling-type;
      description
        "Virtual forwarder interface";
    }
    identity evi {
      base vpn-signaling-type;
      description
        "Ethernet virtual interconnect.";
    }
    identity l2vpn-type {
      description
        "Layer 2 VPN types";
    }
    identity vpws {
      base l2vpn-type;
      description
        "Virtual Private Wire Service";
    }
    identity vpls {
      base l2vpn-type;
      description
        "Virtual Private LAN Service";
    }
    identity evpn {
      base l2vpn-type;
      description
        "Ethernet VPN";
    }
    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 svc-topo-type {
        description
          "Service topology Type";
      }
      identity point-to-point {
        base svc-topo-type;
        description
          "Point to Point.";
      }
      identity multipoint-to-multipoint {
        base svc-topo-type;
        description
          "Multipoint to Multipoint.";
      }
      identity rooted-multipoint {
        base svc-topo-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";
    }

  /* Groupings */

  grouping customer-info-grouping {
    list customer-info {
      key "customer-account-number customer-name";
      leaf customer-account-number {
        type uint32;
```

```
          description
            "Customer account number";
        }
        leaf customer-name {
          type string;
          description
            "Customer name";
        }
        container customer-operation-center {
          leaf customer-noc-street-address {
            type string;
            description
              "Customer NOC street Address.";
          }
          container customer-noc-phone-number {
            leaf main-phone-num {
              type uint32;
              description
                "Main phone number.";
            }
            leaf extension-options {
              type uint32;
              description
                "Extension or options";
            }
            description
              "Configuration of customer NOCc phone number";
          }
          description
            "Configuration of customer operation center";
        }
        description
          "List of customer information";
      }
      description
        "Grouping for customer information";
    }

    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 site-name {
            type string;
            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 managemnt {
        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 filter {
              choice lan {
                case lan-tag {
                  leaf-list lan-tag {
                    type string;
                    description
                      "List of lan-tags to be matched.";
                  }
                }
                description
                  "Choice for LAN matching type";
              }
              description
                "If used, it permit to split site LANs
                 among multiple VPNs.
                 If no filter used, all the LANs will be
                 part of the same VPNs with the same
                 role.";
            }
            container vpn {
              leaf vpn-id {
                type leafref {
                  path "/l2vpn-svc/vpn-services/"+
                       "vpn-svc/vpn-id";
                }
                mandatory true;
                description
                  "Reference to an IPVPN.";
              }
              leaf site-role {
                type identityref {
                  base site-role;
                }
                default any-to-any-role;
                description
                  "Role of the site in the IPVPN.";
              }
              description
                "List of VPNs the LAN is associated to.";
            }
            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 location {
        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
          "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 {
        leaf svlan-id-ethernet-tag {
          type string;
           description
             "SVLAN-ID/Ethernet Tag configurations";
        }
        list cvlan-id-to-evc-map {
          key "evc-id type";
          leaf evc-id {
            type leafref {
              path "/l2vpn-svc/vpn-services/vpn-svc/vpn-id";
            }
            description
              "EVC ID";
          }
          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";
        }
        leaf service-level-mac-limit {
          type string;
          description
            "Service-level MAC-limit (E-LAN only)";
        }
        description
          "This grouping defines site service parameters";
      }

      grouping service-protection {
        container service-protection {
```

```
          container protection-model {
            description
              "Container of protection model configurations";
          }
          container peer-evc-id {
            description
              "Container of peer EVC ID configurations";
          }
          description
            "Container of End-to-end Service Protection
             configurations";
        }
        description
          "Grouping for service protection";
      }

      grouping ethernet-service-type {
        choice ethernet-svc-type {
          case e-line {
            leaf epl {
              type boolean;
              description
                "Ethernet private line";
            }
            leaf evpl {
              type boolean;
              description
                "Ethernet virtual private line";
            }
            description
              "Case of e-line";
          }
          case e-lan {
            leaf ep-lan {
              type boolean;
              description
                "Ethernet private LAN";
            }
            leaf evp-lan {
              type boolean;
              description
                "Ethernet virtual private LAN";
            }
            description
              "Case of e-lan";
          }
          case e-access {
            leaf access-epl {
```

```
            type boolean;
            description
              "Access Ethernet virtual private line";
          }
          leaf access-evpl {
            type boolean;
            description
              "Access Ethernet virtual private line";
          }
          description
            "Case of e-access.";
        }
        description
          "Choice of Ethernet service type";
      }
      description
        "Grouping for Ethernet service type.";
    }

    grouping signaling-option-grouping {
      list signaling-option {
        key "type";
        leaf type {
          type identityref {
            base vpn-signaling-type;
          }
          description
            "VPN signaling types";
        }
        container mp-bgp-l2vpn {
          leaf vpn-id {
            type svc-id;
            description
              "Identifies the target VPN";
          }
          leaf type {
            type identityref {
              base l2vpn-type;
            }
            description
              "L2VPN types";
          }
          description
            "Container for MP BGP L2VPN";
        }
        container mp-bgp-evpn {
          leaf vpn-id {
            type svc-id;
```

```
            description
              "Identifies the target VPN";
          }
          leaf type {
            type identityref {
              base l2vpn-type;
            }
            description
              "L2VPN types";
          }
          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 {
          list PE-EG-list {
            key "service-ip-lo-addr vc-id";
            leaf service-ip-lo-addr {
              type inet:ip-address;
              description
                "Service ip lo address";
            }
            leaf vc-id {
              type string;
              description
              "VC id";
            }
            description
              "List of PE/EG";
          }
          description
            "Container of T-LDP PWE configurations";
        }
        container pwe-encapsulation-type {
          leaf ethernet {
```

```
            type boolean;
            description
              "Ethernet";
          }
          leaf vlan {
            type boolean;
            description
              "VLAN";
          }
          description
            "Container of PWE Encapsulation Type configurations";
        }
        container pwe-mtu {
          leaf allow-mtu-mismatch {
            type boolean;
            description
              "Allow MTU mismatch";
          }
          description
            "Container of PWE MTU configurations";
        }
        container control-word {
          description
            "Container of control word configurations";
        }
        description
          "List of VPN Signaling Option.";
      }
      description
        "Grouping for signaling option";
    }

    grouping load-balance-grouping {
      leaf fat-pw {
        type boolean;
        description
          "Fat label is applied to Pseudowires across MPLS
           network";
      }
      leaf entropy-label {
        type boolean;
        description
          "Entropy label is applied to IP forwarding,
           L2VPN or L3VPN across MPLS network";
      }
      leaf vxlan-source-port {
        type string;
        description
```

```
            "Vxlan source 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 "metro-mkt-id mkt-name";
         leaf metro-mkt-id {
           type uint32;
           description
             "Metro MKT ID";
         }
         leaf mkt-name {
           type string;
           description
             "MKT Name";
         }
         leaf ovc-id {
          type string;
          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 evc-id-grouping {
      leaf evc-id {
        type svc-id;
        description
          "Ethernet Virtual Connection identifier";
      }
      description
        "Grouping for EVC-ID";
    }

    grouping svc-type-grouping {
      container evc-type {
        uses evc-id-grouping;
        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 "network-access-id";
            leaf network-access-id {
              type string;
              description
                "Network Access Identifier";
```

```
            }
            description
              "List for UNIs";
          }
          description
            "Container for UNI List";
        }
        description
          "Container for Ethernet virtual connection.";
      }
      container ovc-type {
        if-feature ovc-type;
        list ovc-list {
          key ovc-id;
          leaf ovc-id {
            type svc-id;
            description
              "OVC ID type";
          }
          leaf on-net {
            type boolean;
            description
              "On net";
          }
          leaf off-net {
            type boolean;
            description
              "Off net";
          }
          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-filtering {
        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;
        uses B-U-M-storm-control-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
            "Mini link";
        }
        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 encapsulation-type {
          type enumeration {
            enum VLAN {
              description
                "VLAN";
            }
            enum ether {
              description
                "Ethernet";
            }
          }
          description
            "Encapsulation type";
        }
        leaf ethertype {
          type string;
          description
            "Ether type";
        }
        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 encapsulation-type {
          type enumeration {
            enum VLAN {
              description
                "VLAN";
            }
            enum Ethernet {
              description
                "Ethernet";
            }
          }
          description
            "Encapsulation-type";
        }
        leaf ethertype {
          type string;
          description
            "Ethertype";
        }
        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 {
          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 l2-access-grouping {
        container dot1q {
          when "'../l2-access-type'='dot1q'";
          leaf physical-inf {
            type string;
            description
              "Physical Interface";
          }
          leaf vlan-id {
            type uint32;
            description
              "VLAN identifier";
          }
          description
            "Qot1q";
        }
        container qinq {
          when "'../l2-access-type'='qinq'";
          leaf s-vlan-id {
            type uint32;
```

```
          description
            "S-VLAN Identifier";
        }
        leaf c-vlan-id {
          type uint32;
          description
            "C-VLAN Identifier";
        }
        description
          "QinQ";
      }
      leaf sub-if-id {
        when "'../l2-access-type'='sub-interface'";
        type uint32;
        description
          "Sub interface ID";
      }
      container vxlan {
        when "'../l2-access-type'='vxlan'";
        leaf vni-id {
           type uint32;
           description
             "VNI Identifier";
        }
        list peer-list {
          key peer-ip;
          leaf peer-ip {
            type inet:ip-address;
            description
              "Peer IP";
          }
          description
            "List for peer IP";
        }
        description
          "QinQ";
      }
      description
        "Grouping for Layer2 access";
    }

    grouping ethernet-connection-grouping {
      container ethernet-connection {
        leaf ESI {
         type string;
          description
          "Ethernet segment id";
        }
```

```
        leaf interface-description {
          type string;
          description
           "Interface description";
        }
        container vlan {
          leaf vlan-id {
            type uint32;
            description
              "VLAN-ID/Ethernet Tag configurations";
          }
          description
            "Abstract container for VLAN";
        }
        uses l2-access-grouping;
        uses phy-interface-grouping;
        uses lag-interface-grouping;
        description
          "Container for bearer";
      }
      description
        "Grouping for bearer.";
    }

    grouping evc-mtu-grouping {
      leaf evc-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 {
           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 {
         leaf stp-rstp-mstp {
           type control-mode;
           description
             "STP/RSTP/MSTP";
         }
         leaf pause {
           type control-mode;
           description
             "Pause";
         }
         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";
          }
          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 service-level-grouping {
      container service-level {
        leaf cos-identifier {
          type identityref {
           base cos-id;
          }
          description
            "COS Identifier [ EVC | EVC + PCP ]";
        }
        leaf color-identifier {
          type identityref {
           base color-id;
          }
          description
            "Color Identifier [ EVC | EVC + CVLAN ]";
        }
        leaf ingress-bw-profile-per-evc {
          type string;
          description
            "Ingress Bandwidth Profile per EVC";
        }
        leaf ingress-bw-profile-per-cos-id {
          type string;
          description
            "Ingress Bandwidth Profile per COS Identifier";
        }
        leaf egress-bw-profile-per-evc {
          type string;
          description
            "Egress Bandwidth Profile per EVC";
        }
        leaf egress-bw-profile-per-cos-id {
          type string;
          description
            "Egress Bandwidth Profile per COS Identifier";
        }
        leaf byte-offset {
          type uint16;
          description
            "For not including extra VLAN tags in the QoS
            calculation";
        }
```

```
          leaf policing {
           type identityref {
            base policing;
           }
           description
             "The policing can be either one-rate,
              two-color (1R2C) or two-rate, three-color (2R3C)";
          }
          leaf perf-tier-opt {
            type identityref {
              base perf-tier-opt;
            }
            description
              "Performance tier option";
          }
          leaf COS {
            type uint32;
            description
              "Class of Service";
          }
          description
            "Container of service level configurations.";
        }
        description
          "Grouping for service level.";
      }

      grouping B-U-M-storm-control-grouping {
        container B-U-M-storm-control {
           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 B-U-M-strom-control configurations";
      }
      description
        "Grouping for B-U-M-strom-control";
    }

    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.";
          }
          uses request-type-profile-grouping;
          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-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.";
            }
          }
          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 "id type";
            leaf id {
              type string;
              description
                "ID";
            }
            leaf type {
              type identityref {
                base bw-type;
              }
              description
                "Bandwidth Type";
            }
            leaf evc-id {
              type svc-id;
              description
                "EVC ID";
            }
            leaf CIR {
              type uint32;
              description
                "Committed Information Rate";
            }
            leaf CBS {
              type uint32;
              description
                "Committed Burst Size";
            }
            leaf EIR {
              type uint32;
              description
                "Excess Information Rate";
            }
            leaf EBS {
              type uint32;
              description
                "Excess Burst Size";
            }
            leaf CM {
              type uint32;
              description
                "Color Mode";
            }
            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 "id type";
              leaf id {
                type string;
                description
                  "ID";
              }
              leaf type {
                type identityref {
                  base bw-type;
                }
                description
                  "Bandwidth Type";
              }
              leaf evc-id {
                type svc-id;
                description
                  "EVC ID";
              }
              leaf CIR {
                type uint32;
                description
                  "Committed Information Rate";
              }
              leaf CBS {
                type uint32;
                description
                  "Committed Burst Size";
              }
              leaf EIR {
                type uint32;
                description
                  "Excess Information Rate";
              }
              leaf EBS {
                type uint32;
                description
                  "Excess Burst Size";
              }
              leaf CM {
                type uint32;
                description
                  "Color Mode";
```

```
          }
          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 cos-color-id {
          leaf device-id {
            type string;
            description
              "Device 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 criterions.";
        }
        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 ingress-profile {
                type string;
                description
                  "Ingress QoS Profile to be used";
              }
              leaf egress-profile {
                type string;
                description
                  "Egress 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 direction-type;
                        description
                          "Direction type";
                      }
                      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 perf-tier-opt {
                        type identityref {
                          base perf-tier-opt;
                        }
                        description
                          "Performance tier option";
                      }
                      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";
                    }
                    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 service-grouping {
        container service {
          uses site-service-basic;
          uses site-service;
          leaf service-multiplexing {
```

```
            type boolean;
            description
              "Service multiplexing";
          }
          uses site-service-qos-profile;
          description
            "Container for service";
        }
        description
          "Grouping for service.";
      }

    /* MAIN L2VPN SERVICE */

    container l2vpn-svc {
      container customer-info {
        uses customer-info-grouping;
        description
          "Container for customer information.";
       }
       container vpn-services {
         list vpn-svc {
           key "vpn-id";
           leaf vpn-id {
             type svc-id;
             description
               "Defining a service id.";
           }
           leaf svc-type {
             type identityref {
               base service-type;
             }
             description
               "Service type";
           }
           uses svc-type-grouping;
           container ethernet-svc-type {
             uses ethernet-service-type;
             description
               "Container of Ethernet service type";
           }
           leaf svc-topo {
             type identityref {
               base svc-topo-type;
             }
             description
               "Defining service topology, such as
                point to point, multipoint to multipoint,
```

```
            rooted multipoint, etc.";
        }
        uses vpn-service-cloud-access; // need fixed??
        container ce-vlan-preservation {
          description
            "CE vlan preservation";
        }
        container metro-network-id {
          uses inter-mkt-service;
          uses intra-mkt-grouping;
          description
            "Container of Metro-Network ID configurations";
        }
        container L2CP-control {
          leaf stp-rstp-mstp {
            type control-mode;
            description
              "STP/RSTP/MSTP";
          }
          leaf pause {
            type control-mode;
            description
              "Pause";
          }
          leaf lldp {
            type boolean;
            description
              "LLDP";
          }
          description
            "Container of L2CP control configurations";
        }
        container load-balance-options {
          uses load-balance-grouping;
          description
            "Container for load balance options";
        }
        uses site-service;
        uses service-protection;
        container sla-targets {
          description
            "Container for SLA targets";
        }
        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 site-management;
            uses customer-location-info;
            uses site-diversity;
            uses site-vpn-policy ;
            container signaling-option {
              if-feature signaling-option;
              uses signaling-option-grouping;
              description
                "Container for signaling option";
            }
            container load-balance-options {
              uses load-balance-grouping;
              description
                "Container for load balance options";
            }
            uses operational-requirements-ops;
            container ports {
              list port {
                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 l2cp-grouping;
              uses evc-mtu-grouping;
              uses availability-grouping;
              uses vpn-attachment-grouping;
              uses service-grouping;
              uses ethernet-svc-oam-grouping;
              uses site-security;
              description
                "List of ports";
            }
            description
              "Container of port configurations";
          }
          description
            "List of sites";
        }
        description
          "Container of site configurations";
      }

      description
        "Container for L2VPN service";
    }
  }
  <CODE ENDS>
```

9. Security Considerations

The YANG 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.

This document has drawn on the work of the L3SM Working Group expressed in [I-D.ietf-l3sm-l3vpn-service-model].

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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://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, <http://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, <http://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, <http://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, <http://www.rfc-editor.org/info/rfc6020>.

[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, <http://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, <http://www.rfc-editor.org/info/rfc6536>.

[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, <http://www.rfc-editor.org/info/rfc7432>.

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

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-01 (work in progress), July 2016.

[I-D.ietf-bess-l2vpn-yang]
Shah, H., Brissette, P., Chen, I., Hussain, I., and B. Wen, "YANG Data Model for MPLS-based L2VPN", draft-ietf-bess-l2vpn-yang-02 (work in progress), October 2016.

[I-D.ietf-l3sm-l3vpn-service-model]
Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN service delivery", draft-ietf-l3sm-l3vpn-service-model-19 (work in progress), November 2016.

[I-D.wu-opsawg-service-model-explained]
Wu, Q., LIU, S., and A. Farrel, "Service Models Explained", draft-wu-opsawg-service-model-explained-05 (work in progress), January 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, <http://www.rfc-editor.org/info/rfc6624>.

Authors' Addresses

Bin Wen
Comcast

Email: bin_wen@comcast.com

Giuseppe Fioccola
Telecom Italia

Email: giuseppe.fioccola@telecomitalia.it

Chongfeng Xie
China Telecom

Email: xiechf@ctbri.com.cn

Luay Jalil
Verizon

Email: luay.jalil@verizon.com

OPS Area Working Group
Internet-Draft
Intended status: Informational
Expires: December 1, 2017

Q. Wu
W. Liu
Huawei Technologies
A. Farrel
Juniper Networks

May 30, 2017

Service Models Explained
draft-wu-opsawg-service-model-explained-06

Abstract

The IETF has produced a considerable number of data modules in the YANG modelling language. The majority of these modules are used to construct data models to model devices or monolithic functions and they allow access for configuration and to read operational status.

A small number of YANG modules have been defined to model services (for example, the Layer Three Virtual Private Network Service Model produced by the L3SM working group and documented in RFC 8049).

This document briefly sets out the scope of and purpose of an IETF service model, and it also shows where a service model might fit into a Software Defined Networking architecture. Note that service models do not make any assumption of how a service is actually engineered and delivered for a customer; details of how network protocols and devices are engineered to deliver a service are captured in other models that are not exposed through the Customer-Provider Interface.

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 http://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 December 1, 2017.

Table of Contents

1. Introduction

In recent years the number of data modules written in the YANG modelling language [RFC6020] for configuration and monitoring has blossomed. Many of these are used for device-level configuration (for example, [RFC7223]) or for control of monolithic functions or protocol instances (for example, [RFC7407]).

Within the context of Software Defined Networking (SDN) [RFC7426] YANG data models may be used on Southbound Interfaces (SBIs) between a controller and network devices, and between network orchestrators and controllers. There may also be a hierarchy of such components with super-controllers, domain controllers, and device controllers all exchanging information and instructions using YANG models.

Recently there has been interest in using YANG to define and document data models that describe services in a portable way that is independent of which network operator uses the model. For example, the Layer Three Virtual Private Network Service Model (L3SM) [RFC8049]. Such models may be used in manual and even paper-driven service request processes with a gradual transition to IT-based mechanisms. Ultimately they could be used in online, software-driven dynamic systems.

This document explains the scope and purpose of service models within the IETF and describes how a service model can be used by a network operator. Equally, this document clarifies what a service model is not, and dispels some common misconceptions.

The document also shows where a service model might fit into an SDN architecture, but it is important to note that a service model does not require or preclude the use of SDN. Note that service models do not make any assumption of how a service is actually engineered and delivered to a customer; details of how network protocols and devices are engineered to deliver a service are captured in other models that are not exposed through the Customer- Provider Interface.

Other work on classifying YANG data models has been done in [I-D.ietf-netmod-yang-model-classification]. That document provides an important reference for this document, and also uses the term "service model". Section 6.1 provides a comparison between these two uses of the same terminology.

2. Terms and Concepts

Readers should familiarize themselves with the description and classification of YANG models provided in [I-D.ietf-netmod-yang-model-classification].

The following terms are used in this document:

Network Operator: This term is used to refer to the company that owns and operates one or more networks that provide Internet connectivity services and/or other services. The term is also used to refer to an individual who performs operations and management on those networks.

Customer: This term refers to someone who purchases a service (including connectivity) from a network operator. In the context of this document, a customer is usually a company that runs their own network or computing platforms and wishes to connect to the Internet or between sites. Such a customer may operate an enterprise network or a data center. Sometimes this term may also be used to refer to the individual in such a company who contracts to buy services from a network operator. A customer as described here is a separate commercial operation from the network operator, but some companies may operate with internal customers so that, for example, an IP/MPLS packet network may be the customer of an optical transport network.

Service: A network operator delivers one or more services to a customer. A service in the context of this document (sometimes called a Network Service) is some form of connectivity between customer sites and the Internet, or between customer sites across the network operator's network and across the Internet. However, a distinction should be drawn between the parameters that describe a service as included in a customer service model (q.v.) and a Service Level Agreement (SLA) as discussed in Section 5 and Section 7.2.

A service may be limited to simple connectivity (such as IP-based Internet access), may be a tunnel (such as a virtual circuit), or may be a more complex connectivity model (such as a multi-site virtual private network). Services may be further enhanced by additional functions providing security, load-balancing, accounting, and so forth. Additionally, services usually include guarantees of quality, throughput, and fault reporting.

This document makes a distinction between a service as delivered to a customer (that is, the service as discussed on the interface between a customer and the network operator) and the service as realized within the network (as described in [I-D.ietf-netmod-yang-model-classification]). This distinction is discussed further in Section 6.

Readers may also refer to [RFC7297] for an example of how an IP connectivity service may be characterized.

Data Model: The concepts of information models and data models are described in [RFC3444]. That document defines a data model by contrasting it with the definition of an information model, so it may be helpful to quote some text to give context within this document.

The main purpose of an information model is to model managed objects at a conceptual level, independent of any specific implementations or protocols used to transport the data. The degree of specificity (or detail) of the abstractions defined in the information model depends on the modeling needs of its designers. In order to make the overall design as clear as possible, an information model should hide all protocol and implementation details. Another important characteristic of an information model is that it defines relationships between managed objects.

Data models, conversely, are defined at a lower level of abstraction and include many details. They are intended for implementors and include protocol-specific constructs.

Service Model: A service model is a specific type of data model. It describes a service and the parameters of the service in a portable way. The service model may be divided into two categories:

Customer Service Model: A customer service model is used to describe a service as offered or delivered to a customer by a network operator. It can be used by a human (via a user interface such as a GUI, web form, or CLI) or by software to configure or request a service, and may equally be consumed by a human (such as via an order fulfillment system) or by a software component. Such models are sometimes referred to simply as "service models" [RFC8049]. A customer service model is expressed as a core set of parameters that are common across network operators: additional features that are specific to the offerings of individual network operators would be defined in extensions or augmentations of the model. Except where specific technology details (such as encapsulations, or mechanisms applied on access links) are directly pertinent to the customer, customer service models are technology agnostic so that the customer does have influence over or knowledge of how the network operator engineers the service.

An example of where such details are relevant to the customer are when they describe the behavior or interactions on the interface between the equipment at the customer site (often referred to as the Customer Edge or CE equipment) and the equipment at the network operator’s site (usually referred to as the Provider Edge or PE equipment).

Service Delivery Model: A service delivery model is used by a network operator to define and manage how a service is engineered in the network. It can be used by a human operator (such as via a management station) or by a software tool to instruct network components. Such models are sometimes referred to as "network service models" [I-D.ietf-netmod-yang-model-classification] and are consumed by "external systems" such as Operations Support System (OSS). A service delivery model is expressed as a core set of parameters that are common across a network type and technology: additional features that are specific to the configuration of individual vendor equipment or proprietary protocols would be defined in extensions or augmentations of the model. Service delivery models include technology-specific modules.

The distinction between a customer service model and a service delivery model needs to be repeatedly clarified. A customer service model is not a data model used to directly configure network devices, protocols, or functions: it is not something that is sent to network devices (i.e., routers or switches) for processing. Equally, a customer service model is not a data model that describes how a network operator realizes and delivers the service described by the model. This distinction is discussed further in later sections.

3. Using Service Models

As already indicated, customer service models are used on the interface between customers and network operators. This is shown simply in Figure 1

The language in which a customer service model is described is a choice for whoever specifies the model. The IETF uses the YANG data modeling language defined in [RFC6020]

The encoding and communication protocol used to exchange a customer service model between customer and network operator are deployment- and implementation-specific. The IETF has standardized the NETCONF protocol [RFC6241] and the RESTCONF protocol [RFC8040] for interactions "on the wire" between software components with data encoded in XML or JSON. However, co-located software components might use an API, while systems with more direct human interactions might use web pages or even paper forms.

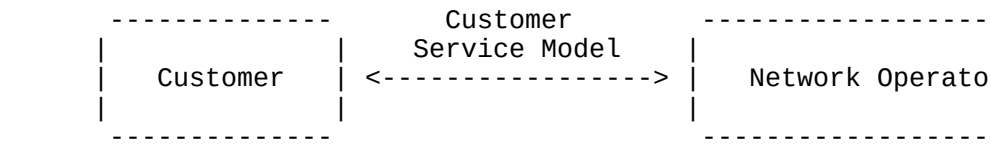

Figure 1: The Customer Service Models used on the Interface between Customers and Network Operators

How a network operator processes a customer's service request described with a customer service model depends on the commercial and operational tools, processes, and policies used by the network operator. These may vary considerably from one network operator to another.

However, the intent is that the network operator maps the service request into configuration and operational parameters that control one or more networks to deliver the requested services. That means that the network operator (or software run by the network operator) takes the information in the customer service model and determines how to deliver the service by enabling and configuring network protocols and devices. They may achieve this by constructing service delivery models and passing them to network orchestrators or controllers. The use of standard customer service models eases service delivery by means of automation.

The practicality of customer service models has been repeatedly debated. It has been suggested that network operators have such radically different business modes and such diverse commercial offerings that a common customer service model is impractical. However, the L3SM [RFC8049] results from the consensus of multiple individuals working at network operators and offers a common core of service options that can be augmented according to the needs of individual network operators.

It has also been suggested that there should be a single, base customer service module, and that details of individual services should be offered as extensions or augmentations of this. It is quite possible that a number of service parameters (such as the identity and postal address of a customer) will be common and it would be a mistake to define them multiple times, once in each customer service model. However, the distinction between a 'module' and a 'model' should be considered at this point: modules are how the data for models is logically broken out and documented especially for re-use in multiple models.

4. Service Models in an SDN Context

In an SDN system, the management of network resources and protocols is performed by software systems that determine how best to utilize the network. Figure 2 shows a sample architectural view of an SDN system where network elements are programmed by a component called an "SDN controller" (or "controller" for short), and where controllers are instructed by an orchestrator that has a wider view of the whole of, or part of, a network. The internal organization of an SDN control plane is deployment-specific.

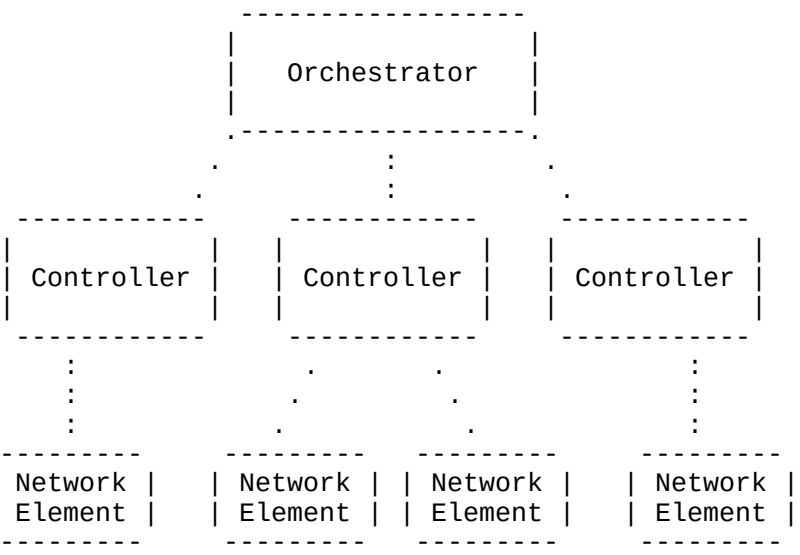

Figure 2: A Sample SDN Architecture

But a customer's service request is (or should be) technology-agnostic. That is, there should be an independence between the behavior and functions that a customer requests and the technology that the network operator has available to deliver the service. This means that the service request must be mapped to the orchestrator's view, and this mapping may include a choice of which networks and technologies to use depending on which service features have been requested.

One implementation option to achieve this mapping is to split the orchestration function between a "Service Orchestrator" and a "Network Orchestrator" as shown in Figure 3.

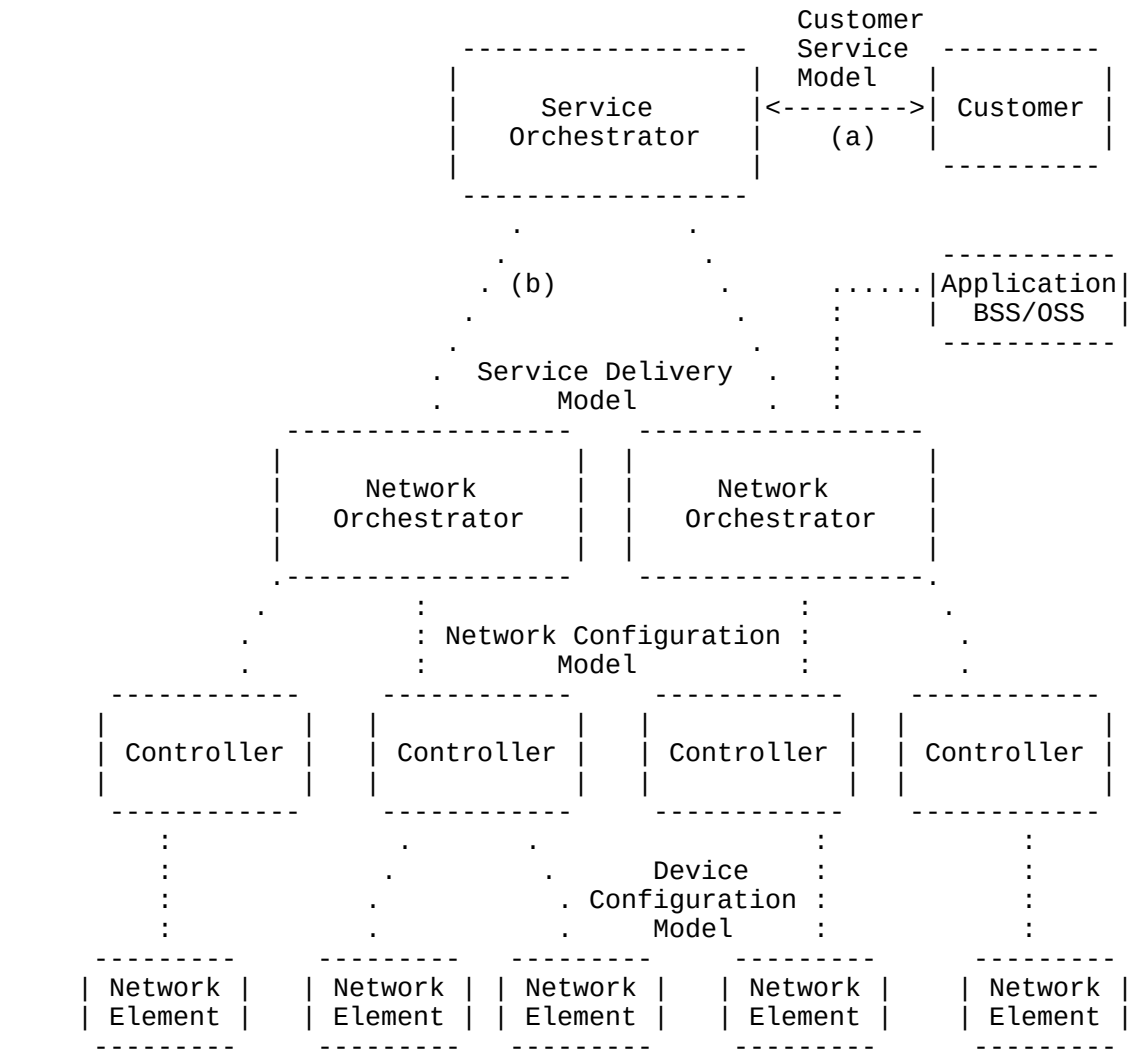

Figure 3: An Example SDN Architecture with a Service Orchestrator

Figure 3 also shows where different data models might be applied within the architecture.

The split between control components that exposes a "service interface" is present in many figures showing extended SDN architectures:

- Figure 1 of [RFC7426] shows a separation of the "Application Plane", the "Network Services Abstraction Layer (NSAL)", and the "Control Plane". It marks the "Service Interface" as situated between the NSAL and the Control Plane.

- [RFC7491] describes an interface between an "Application Service Coordinator" and an "Application-Based Network Operations Controller".

- Figure 1 of [I-D.ietf-netmod-yang-model-classification] shows an interface from an OSS or a Business Support System (BSS) that is expressed in "Network Service YANG Models".

This can all lead to some confusion around the definition of a "service interface" and a "service model". Some previous literature considers the interface northbound of the Network Orchestrator (labeled "(b)" in Figure 3) to be a "service interface" used by an application, but the service described at this interface is network-centric and is aware of many features such as topology, technology, and operator policy. Thus, we make a distinction between this type of service interface and the more abstract service interface (labeled "(a)" in Figure 3) where the service is described by a service model and the interaction is between customer and network operator. Further discussion of this point is provided in Section 5.

5. Possible Causes of Confusion

In discussing service models, there are several possible causes of confusion:

- The services we are discussing are services provided by network operators to customers. This is a completely different thing to "Foo as a Service" (for example, Infrastructure as a Service (IaaS)) where a service provider offers a service at some location that is reached across a network. The confusion arises not only because of the use of the word "service", but also because network operators may offer value-added services as well as network connection services to their customers.

- Network operation is completely out of scope in the discussion of services between a network operator and a customer. That means that the customer service model does not reveal to the customer anything about how the network operator delivers the service. The model does not expose details of technology or network resources used to provide the service. For example, in the simple case of point-to-point virtual link connectivity provided by a network tunnel (such as an MPLS pseudowire) the network operator does not expose the path through the network that the tunnel follows. Of

course, this does not preclude the network operator from taking guidance from the customer (such as to avoid routing traffic through a particular country) or from disclosing specific details (such as might be revealed by a route trace), but these are not standard features of the service as described in the customer service model.

- The network operator may use further data models (service delivery models) that help to describe how the service is realized in the network. These models might be used on the interface between the Service Orchestrator and the Network Orchestrator as shown in Figure 3 and might include many of the pieces of information from the customer service model alongside protocol parameters and device configuration information. [I-D.ietf-netmod-yang-model-classification] also terms these data models as "service models" or "Network Service YANG Models" and a comparison is provided in Section 6.1. It is important that the Service Orchestrator should be able to map from a customer service model to these service delivery models, but they are not the same things.

- Commercial terms are generally not a good subject for standardization. It is possible that some network operators will enhance standard customer service models to include commercial information, but the way this is done is likely to vary widely between network operators.

- Service Level Agreements (SLAs) have a high degree of overlap with the definition of services present in customer service models. Requests for specific bandwidth, for example, might be present in a customer service model, and agreement to deliver a service is a commitment to the description of the service in the customer service model. However, SLAs typically include a number of fine-grained details about how services are allowed to vary, by how much, and how often. SLAs are also linked to commercial terms with penalties and so forth, and so are also not good topics for standardization.

  If a network operator chooses to express an SLA using a data model, that model might be referenced as an extension or an augmentation of the customer service model.

6. Comparison With Other Work

Other work has classified YANG models, produced parallel architectures, and developed a range of YANG models. This section briefly examines that other work and shows how it fits with the description of service models introduced in this document.

6.1. Comparison With Network Service Models

As previously noted, [I-D.ietf-netmod-yang-model-classification] provides a classification of YANG data models. It introduces the term "Network Service YANG Module" to identify the type of model used to "describe the configuration, state data, operations and notifications of abstract representations of services implemented on one or multiple network elements." These are service delivery models as described in this document, that is, they are the models used on the interface between the Service Orchestrator or OSS/BSS and the Network Orchestrator as shown in Figure 3.

Figure 1 of [I-D.ietf-netmod-yang-model-classification] can be modified to make this more clear and to add an additional example of a Network Service YANG model as shown in Figure 4.

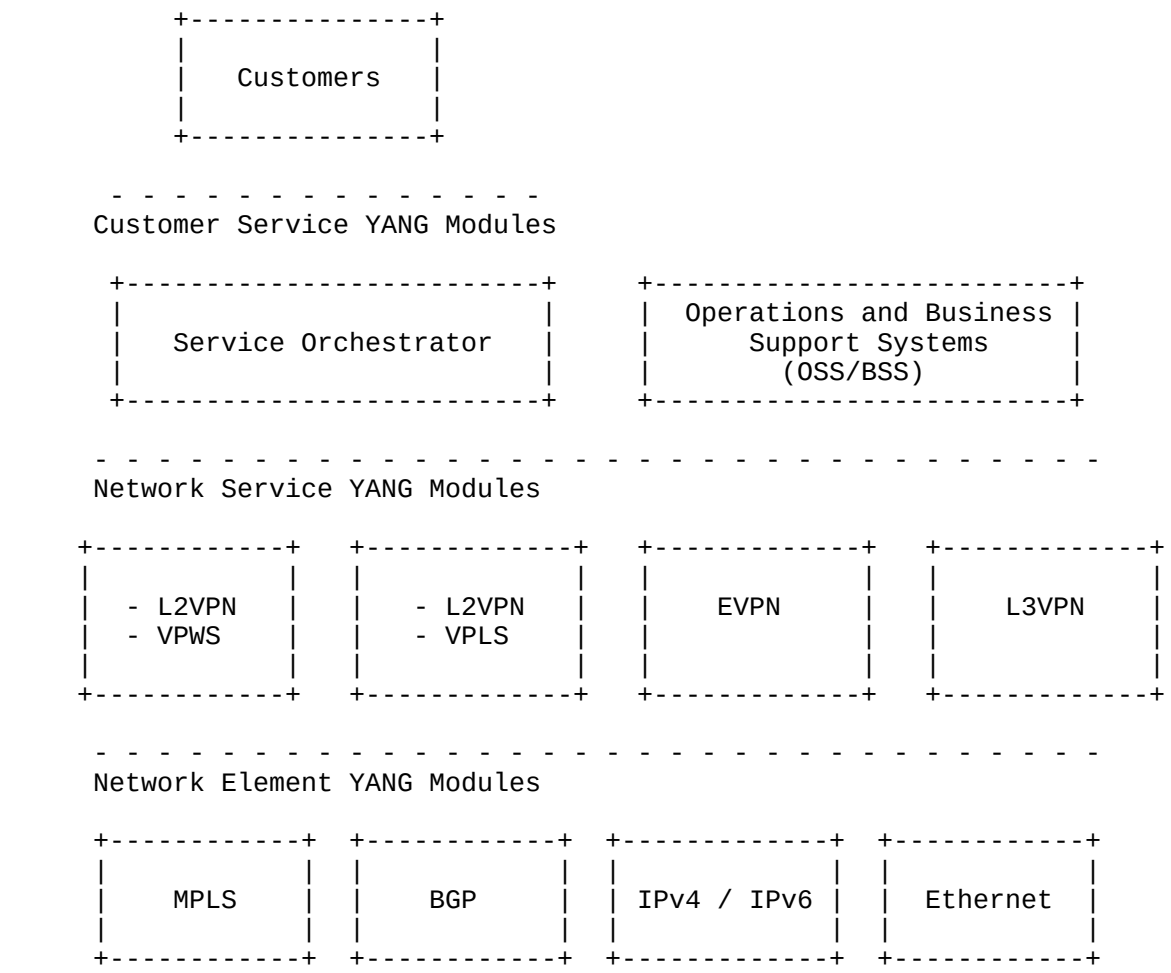

Figure 4: YANG Module Layers Showing Service Models

6.2. Service Delivery and Network Element Model Work

A number of IETF working groups are developing YANG models related to services. These models focus on how the network operator configures the network through protocols and devices to deliver a service. Some of these models are classed as service delivery models while others

have details that are related to specific element configuration and so are classed as network element models.

A sample set of these models is listed here:

- [I-D.dhjain-bess-bgp-l3vpn-yang] defines a YANG model that can be used to configure and manage BGP Layer 3 VPNs.

- [I-D.ietf-bess-l2vpn-yang] documents a YANG model that it is expected will be used by the management tools run by the network operators in order to manage and monitor the network resources that they use to deliver L2VPN services.

- [I-D.ietf-bess-evpn-yang] defines YANG models for delivering an Ethernet VPN service.

6.3. Customer Service Model Work

Several initiatives within the IETF are developing customer service models. The most advanced presents the Layer Three Virtual Private Network (L3VPN) service as described by a network operator to a customer. This L3VPN service model (L3SM) is documented in [RFC8049] where its usage is described as in Figure 5 which is reproduced from that document. As can be seen, the L3SM is a customer service model as described in this document.

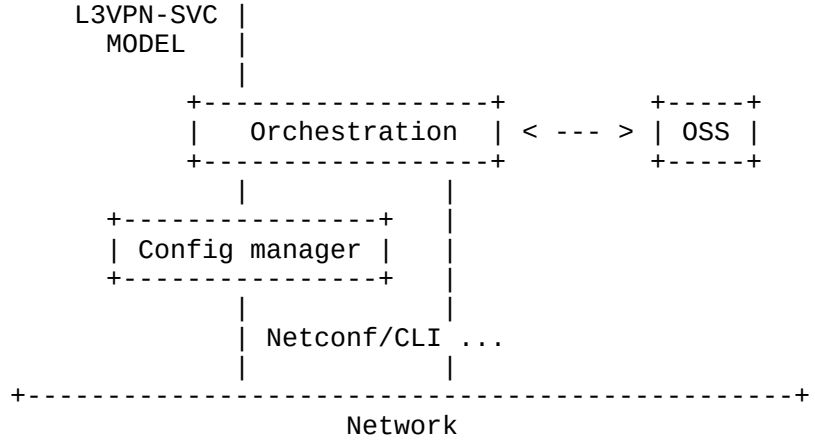

Figure 5: The L3SM Service Architecture

A Layer Two VPN service model (L2SM) is defined in [I-D.ietf-l2sm-l2vpn-service-model]. That model's usage is described

as in Figure 6 which is a reproduction of Figure 5 from that document. As can be seen, the L2SM is a customer service model as described in this document.

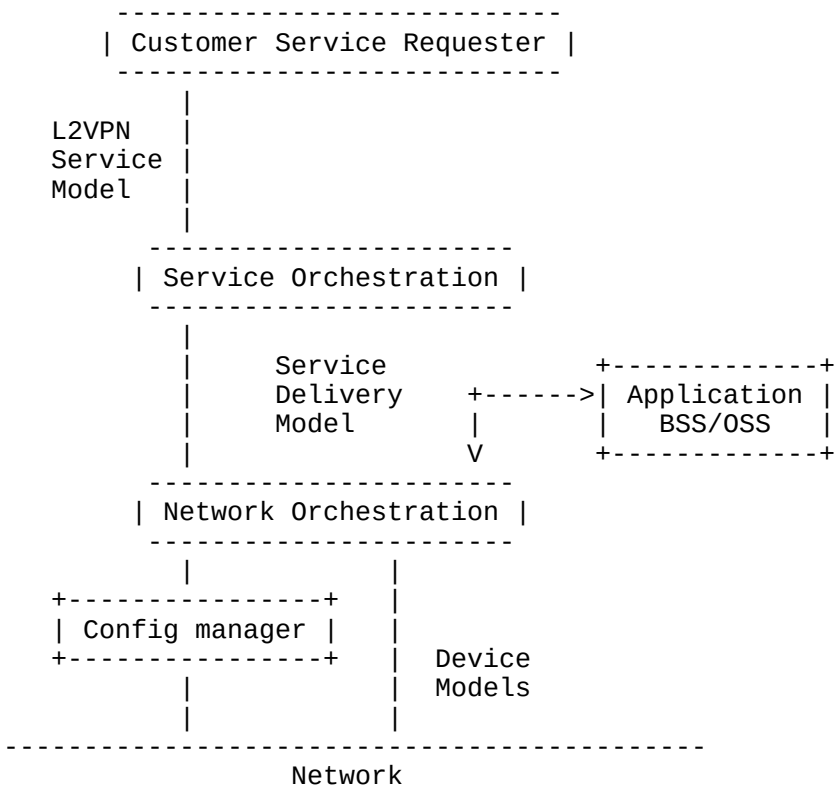

Figure 6: The L2SM Service Architecture

6.4. The MEF Architecture

The MEF Forum has developed an architecture for network management and operation. It is documented as the Lifecycle Service Orchestration (LSO) Reference Architecture and illustrated in Figure 2 of [MEF-55].

The work of the MEF Forum embraces all aspects of Lifecycle Service Orchestration including billing, SLAs, order management, and life-cycle management. The IETF's work on service models is typically smaller offering a simple, self-contained service YANG module. Thus, it may be impractical to fit IETF service models into the MEF Forum

LSO architecture. This does not invalidate either approach, but only observes that they are different.

7. Further Concepts

This section introduces a few further, more advanced concepts

7.1. Technology Agnostic

Service models should generally be technology agnostic. That is to say, the customer should not care how the service is provided so long as the service is delivered.

However, some technologies reach the customer site and make a difference to the type of service delivered. Such features do need to be described in the service model.

Two examples are:

- The data passed between customer equipment and network operator equipment will be encapsulated in a specific way, and that data plane type forms part of the service.

- Protocols that are run between customer equipment and network operator equipment (for example, Operations, Administration, and Maintenance protocols, protocols for discovery, or protocols for exchanging routing information) need to be selected and configured as part of the service description.

7.2. Relationship to Policy

Policy appears as a crucial function in many places during network orchestration. A Service Orchestrator will, for example, apply the network operator’s policies to determine how to provide a service for a particular customer (possibly considering commercial terms). However, the policies within a service model are limited to those over which a customer has direct influence and that are acted on by the network operator.

The policies that express desired behavior of services on occurrence of specific events are close to SLA definitions: they should only be included in the base service model where they are common to all network operators’ offerings. Policies that describe who at a customer may request or modify services (that is, authorization) are close to commercial terms: they, too, should only be included in the base service model where they are common to all network operators’ offerings.

Nevertheless, policy is so important that all service models should be designed to be easily extensible to allow policy components to be added and associated with services as needed.

7.3. Operator-Specific Features

When work in the L3SM working group was started, there was some doubt as to whether network operators would be able to agree on a common description of the services that they offer to their customers because, in a competitive environment, each markets the services in a different way with different additional features. However, the working group was able to agree on a core set of features that multiple network operators were willing to consider as "common". They also understood that should an individual network operator want to describe additional features (operator-specific features) they could do so by extending or augmenting the L3SM model.

Thus, when a basic description of a core service is agreed and documented in a service model, it is important that that model should be easily extended or augmented by each network operator so that the standardized model can be used in a common way and only the operator-specific features varied from one environment to another.

7.4. Supporting Multiple Services

Network operators will, in general, offer many different services to their customers. Each would normally be the subject of a separate service model.

It is an implementation and deployment choice whether all service models are processed by a single Service Orchestrator that can coordinate between the different services, or whether each service model is handled by a specialized Service Orchestrator able to provide tuned behavior for a specific service.

It is expected that, over time, certain elements of the service models will be seen to repeat in each model. An example of such an element is the postal address of the customer.

It is anticipated that, while access to such information from each service model is important, the data will be described in its own module and may form part of the service model either by inclusion or by index.

8. Security Considerations

The interface between customer and service provider is a commercial interface and needs to be subject to appropriate confidentiality. Additionally, knowledge of what services are provided to a customer or delivered by a network operator may supply information that can be used in a variety of security attacks.

Clearly, the ability to modify information exchanges between customer and network operator may result in bogus requests, unwarranted billing, and false expectations. Furthermore, in an automated system, modifications to service requests or the injection of bogus requests may lead to attacks on the network and delivery of customer traffic to the wrong place.

Therefore it is important that the protocol interface used to exchange service request information between customer and network operator is subject to authorization, authentication, and encryption. This document discusses modeling that information, not how it is exchanged.

9. Manageability Considerations

This whole document discusses issues related to network management.

It is important to observe that automated service provisioning resulting from use of a customer service model may result in rapid and significant changes in traffic load within a network and that that might have an effect on other services carried in a network.

It is expected, therefore, that a Service Orchestration component has awareness of other service commitments, that the Network Orchestration component will not commit network resources to fulfill a service unless doing so is appropriate, and that a feedback loop will be provided to report on degradation of the network that will impact the service.

The operational state of a service does not form part of a customer service model. However, it is likely that a network operator may want to report some state information about various components of the service, and that could be achieved through extensions to the core service model.

10. IANA Considerations

This document makes no requests for IANA action

11. Acknowledgements

Thanks to Daniel King, Xian Zhang, and Michael Scharf for useful review and comments. Med Boucadair gave thoughtful and detailed comments on version -04 of this document. Thanks to Dean Bogdanovic and Tianran Zhou for their help coordinating with [I-D.ietf-netmod-yang-model-classification].

12. References

12.1. Normative References

[I-D.ietf-netmod-yang-model-classification]
Bogdanovic, D., Claise, B., and C. Moberg, "YANG Module Classification", draft-ietf-netmod-yang-model-classification-07 (work in progress), May 2017.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <http://www.rfc-editor.org/info/rfc3444>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <http://www.rfc-editor.org/info/rfc7426>.

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

12.2. Informative References

[I-D.dhjain-bess-bgp-l3vpn-yang]
Jain, D., Patel, K., Brissette, P., Li, Z., Zhuang, S., Liu, X., Haas, J., Esale, S., and B. Wen, "Yang Data Model for BGP/MPLS L3 VPNs", draft-dhjain-bess-bgp-l3vpn-yang-02 (work in progress), August 2016.

[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-05 (work in progress), March 2017.

[I-D.ietf-l2sm-l2vpn-service-model]
Wen, B., Fioccola, G., Xie, C., and L. Jalil, "A YANG Data Model for L2VPN Service Delivery", draft-ietf-l2sm-l2vpn-service-model-01 (work in progress), May 2017.

[MEF-55] MEF Forum, "Service Operations Specification MEF 55 : Lifecycle Service Orchestration (LSO) Reference Architecture and Framework", March 2016.

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

[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, <http://www.rfc-editor.org/info/rfc6241>.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <http://www.rfc-editor.org/info/rfc7223>.

[RFC7297] Boucadair, M., Jacquenet, C., and N. Wang, "IP Connectivity Provisioning Profile (CPP)", RFC 7297, DOI 10.17487/RFC7297, July 2014, <http://www.rfc-editor.org/info/rfc7297>.

[RFC7407] Bjorklund, M. and J. Schoenwaelder, "A YANG Data Model for SNMP Configuration", RFC 7407, DOI 10.17487/RFC7407, December 2014, <http://www.rfc-editor.org/info/rfc7407>.

[RFC7491] King, D. and A. Farrel, "A PCE-Based Architecture for Application-Based Network Operations", RFC 7491, DOI 10.17487/RFC7491, March 2015, <http://www.rfc-editor.org/info/rfc7491>.

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

Authors' Addresses

Qin Wu
Huawei Technologies

Email: bill.wu@huawei.com

Will Liu
Huawei Technologies

Email: liushucheng@huawei.com

Adrian Farrel
Juniper Networks

Email: afarrel@juniper.net