A Layer 2 VPN Network Yang Model
draft-barguil-opsawg-l2sm-l2nm-00

Abstract

This document defines a Yang Data model, L2NM, that can be used to manage the provisioning of Layer 2 VPN services in a Service Provider Network. The Yang module defined in this document provides representation of the Layer 2 VPN Service and is aimed at being used by a Network Controller, which can derive from it the configuration information that will be sent to the network devices, as well the the underlay transport network in a service provider environment.

RFC 8466 defines a Layer 2 VPN service model (L2SM) that can be used in the communication between a Customer and the Network Operator. This document complements L2SM by creating a network-centric view of the service to be used in the communication between the entity that interacts directly with the customer, the service orchestrator, (either fully automated or a human operator) and the entity in charge of network orchestration and control (a.k.a., network controller/orchestrator).

The YANG data model defined in this document conforms to the Network Management Datastore Architecture defined in RFC 8342.

Status of This Memo

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

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

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

This Internet-Draft will expire on June 7, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ........................................ 3
   1.1. Terminology .................................... 3
   1.2. Requirements Language ............................ 4
2. Reference architecture ............................... 4
3. Description of the L2NM YANG Module ............... 8
   3.1. Structure of the Module ......................... 9
   3.2. L2 VPN Service ................................... 9
       3.2.1. L2 VPN Service Types ....................... 10
       3.2.2. Underlying Transport Selection .......... 10
       3.2.3. Other relevant Network Parameters ........ 11
   3.3. L2NM Module Tree ................................ 12
4. Relation with other Yang Models .................... 19
   4.1. Relation with L2SM ............................. 20
   4.2. Relation with Network Topology ................. 20
   4.3. Relation with Device Models .................... 20
5. Yang Module ......................................... 20
6. Acknowledgements .................................... 104
7. Contributors ......................................... 104
8. IANA Considerations ................................ 104
9. Security Considerations ............................. 104
10. References .......................................... 105
    10.1. Normative References .......................... 105
    10.2. Informative References ....................... 106
Authors’ Addresses .................................... 107
1. Introduction

[RFC8466] defines a L2VPN Service Model (L2SM) YANG data model that can be used to communicate customers and network operators. This document complements L2SM by creating a Service Provider network-centric view of the service which can be exposed by a Network Controller of the Service Provider Network. Hence, the model can be used in the communication between the entity that interacts directly with the customer, the service orchestrator, (either fully automated or a human operator) and the entity in charge of network orchestration and control (a.k.a., network controller/orchestrator).

The data model defined in this document is called the L2VPN Network Model (L2NM), playing the role of Service Delivery model (fig 3. of [RFC8466]). It enables further capabilities, such as operational parameters, transport protocols selection and precedence. It also serves as a multi-domain orchestration interface, because this model can transport resources (i.e., VCID) between domains. The data model keeps minimum customer-related information.

This document does not obsolete, but complements, the definitions in [RFC8466]. It aims to provide a broader scope for the L2SM and extend the required network-centric parameters to deploy services over any service provider architecture. The Yang module has reused containers from L2SM for the cases that the requirements from L2SM can be translated directly to L2NM. When groupings were available, they were directly reused, avoiding the redefinition.

1.1. Terminology

This document assumes that the reader is familiar with the contents of [RFC6241], [RFC7950], [RFC8466], [RFC8309], and [RFC8453] and uses terminology from those documents. Tree diagrams used in this document follow the notation defined in [RFC8340].

This document makes use of the following terms:

- L2 VPN Customer Service Model (L2SM): Describes the requirements of a L2 VPN that interconnects a set of sites from the point of view of the customer. The customer service model does not provide details on the Service Provider Network. The L2 VPN Customer Service model is defined in [RFC8466].

- L2 VPN Service Network Model (L2NM): A YANG module that describes a L2 VPN Service in the Service Provider Network. It contains information of the Service Provider network and might include allocated resources. It can be used by network controllers to manage and control the VPN Service configuration in the Service
Provider network. The YANG module can be consumed by a Service Orchestrator to request a VPN Service to a Network controller.

- Service Orchestrator: A functional entity that interacts with the customer of a L2 VPN. The Service Orchestrator interacts with the customer using L2SM. The Service Orchestrator is responsible of the CE-PE attachment circuits, the PE selection, and requesting the L2 VPN service to the network controller.

- Network Controller: A functional entity responsible for the control and management of the service provider network.

- VPN node (vpn-node): An abstraction that represents a set of policies applied to a PE and that belong to a single VPN service (vpn-service). A vpn-service involves one or more vpn-nodes. The vpn-node will identify the Service Provider node in which the VPN is deployed.

- VPN network access (vpn-network-access): An abstraction that represents the network interfaces that are associated to a given vpn-node. Traffic coming from the vpn-network-access belongs to the VPN. The attachment circuits (bearers) between CEs and PEs are terminated in the vpn-network-access.

- VPN Site (vpn-site): A VPN customer’s location that is connected to the Service Provider network via a CE-PE link, which can access at least one VPN.

- VPN Service Provider (SP): A Service Provider offers VPN-related services.

- Service Provider (SP) Network: A network able to provide VPN-related services.

1.2. Requirements Language

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

2. Reference architecture

Figure 1 shows the utilization of L2NM in a management stack. The Figure 1 is an expansion of the architecture presented in Section 3 of [RFC8466] and decomposes the box marked "orchestration" in that
figure into three separate functional components called "Service Orchestration", "Network Orchestration", and "Domain Orchestration".

Figure 1 uses the terminology from [RFC8309] is introduced to show the distinction between the "Customer Service Model", the "Service Delivery Model", the "Network Configuration Model", and the "Device Configuration Model". In that context, the "Domain Orchestration" and "Config Manager" roles may be performed by "SDN Controllers".
Figure 1: L2SM and L2NM
Figure 2 shows the L2SM and L2NM may also be set in the context of the ACTN architecture [RFC8453]. Figure 2 shows the Customer Network Controller (CNC), the Multi-Domain Service Coordinator (MDSC), and the Provisioning Network Controller (PNC). It also shows the interfaces between these functional units: the CNC-MDSC Interface (CMI), the MDSC-PNC Interface (MPI), and the Southbound Interface (SBI).
Figure 2: L2SM and L2NM in the Context of ACTN

3. Description of the L2NM YANG Module

The L2NM module (‘ietf-l2vpn-ntw’) is meant to manage L2 VPNs in a service provider network. In particular, the ‘ietf-l2vpn-ntw’ module can be used to create, modify, and retrieve L2VPN Services in a
Network Controller. The module is not aimed at maintaining customer-related information.

Editor’s note: Next version of the document will include the full description of the parameters. When the parameters match with L2SM, the exact reference will be done

3.1. Structure of the Module

The ‘ietf-l3vpn-ntw’ module uses two main containers: ‘vpn-services’ and ‘vpn-profiles’. The ‘vpn-services’ container maintains the set of L2 VPN Services managed in the service provider network. The module allows to create a new L2 VPN service by adding a new instance of ‘vpn-service’. The ‘vpn-service’ is the data structure that abstracts the VPN Service.

```
module: ietf-l2vpn-ntw
    +--rw l2vpn-ntw
        +--rw vpn-profiles
            .......
        +--rw vpn-services
            +--rw vpn-service* [vpn-id]
                .......
```

Figure 3

3.2. L2 VPN Service

The ‘vpn-svc’ is the data structure that abstracts a L2 VPN Service in the Service Provider Network. Every ‘vpn-svc’ has a unique identifier: vpn-id. Such vpn-id is only meaningful locally within the Network controller. In order to facilitate the recognition of the service, a ‘customer-name’ and a ‘description’ may be included. The topology of the VPN service is expressed in the ‘vpn-service-topology’ leaf.

A VPN Service is built by adding instances of ‘vpn-node’ to the ‘vpn-nodes’ container. The ‘vpn-node’ is an abstractions that represent a set of policies applied to a network node and that belong to a single ‘vpn-service’. A ‘vpn-node’ contains ‘vpn_network_accesses’, which are the interfaces involved in the creation of the VPN. The customer sites are connected to the ‘vpn_network_accesses’. Note that, as this is a network data model, the information about customers site is not needed. Such information, is relevant in the L2SM model.
Figure 4

### 3.2.1. L2 VPN Service Types

The L2 VPN Service types directly matches with the L2VPN Service types defined in section 5.1.3 of [RFC8466]:

- Point-to-point VPWSs.
- Point-to-point or point-to-multipoint VPWSs [RFC8214].
- Multipoint VPLSs.
- Multipoint VPLSs connecting one or more root sites and a set of leaf sites but preventing inter-leaf-site communication.
- EVPN services [RFC7432].
- EVPN VPWSs between two customer sites or a set of customer sites as specified in [RFC8214].

### 3.2.2. Underlying Transport Selection

The model enables network operators to select the type of transport protocol underlay. Also, in scenarios with multiple domains and NNI types, the selection of the transport protocol underlay is required. The Service Provider Network might have several underlay possibilities available. If no underlay transport protocol is
specified, the Network Controller will take care of the transport
decision. The following options are supported in the "transport-
encapsulation" container:

- **LDP:** MPLS with LDP (martini encapsulation).
- **GRE:** A mesh of GRE tunnels is established between vpn_nodes.
- **BGP:** BGP tunnels (kompella encapsulation) are preferred to route
traffic between vpn nodes.
- **TE:** TE tunnels (either RSVP-TE or SR) are preferred. The mapping
details will be specified in [draft-ietf-te-service-mapping].
- **SR:** Non-TE SR is preferred to route traffic.

### 3.2.3. Other relevant Network Parameters

This draft is focused in the development of the network centric set of
parameters to provision L2VPNs. Thus, the covered scenarios include
the augmentation of Virtual Circuit Identifier synchronization in a
multidomain environment, precedence management (i.e., Primary and
backup coordination), Split Horizon group management. The parameters
are described by the container associated:

The set of augmentations done in the VPN service are:

- **Status:** Request and modify the service status

The set of augmentations done in the Site Network Access are:

- **Site Bearer:** Reference the NEs and Ports were each site Network
  access belongs. This may help to preserve the local and farends
  from the Network controller site.
- **Precedence:** When two L2VPNs are created at the same time, the
  primary secondary option is enabled according to the tunnel role
  on the network.
- **Lag Interface Support.**
- **Split Horizon group:** Split-horizon group name is extended as part
  of the Site Network Access container.
- **Status:** Request and modify the interface status.
3.3. L2NM Module Tree

The tree structure of the Yang module proposed by this document is shown below:

module: ietf-l2vpn-ntw
  +--rw 12vpn-ntw
    +--rw vpn-profiles
      |   +--rw valid-provider-identifiers
      |     +--rw qos-profile-identifier* [id]
      |        +--rw id string
    +--rw vpn-services
      +--rw vpn-svc* [vpn-id]
        +--rw vpn-id svc-id
        +--rw vpn-svc-type? identityref
        +--rw customer-name? string
        +--rw svc-topo? identityref
        +--rw status
          |   +--rw admin-enabled? boolean
          |   +--ro oper-status? operational-type
          +--rw multicast-like {multicast-like}?
            +--rw enabled? boolean
            +--rw customer-tree-flavors
              |   +--rw tree-flavor* identityref
            +--rw bum-frame-delivery-modes
              |   +--rw bum-frame-delivery* [traffic-type]
              |     +--rw traffic-type identityref
              |     +--rw delivery-mode? identityref
            +--rw multicast-gp-port-mapping? identityref
            +--rw extranet-vpns {extranet-vpn}?
              |   +--rw extranet-vpn* [vpn-id]
              |     +--rw vpn-id svc-id
              |     +--rw local-sites-role? identityref
            +--rw svc-mtu? uint32
            +--rw ce-vlan-preservation? boolean
            +--rw ce-vlan-cos-perservation? boolean
            +--rw transport-encapsulation
              |   +--rw protocol? protocols-type
        +--rw vpn-nodes
          +--rw vpn-node* [vpn-node-id ne-id]
            +--rw vpn-node-id string
            +--rw description? string
            +--rw node-role? identityref
            +--rw ne-id string
          +--rw vpn-network- accesses
            +--rw vpn-network-access* [network-access-id]
              +--rw network-access-id string
++rw description?
    |   string
++rw remote-carrier-name?
    |   string
++rw Interface-mtu?
    |   uint32
++rw status
    |   ++rw admin-enabled?   boolean
    |   ++ro oper-status?   operational-type
++rw access-diversity {site-diversity}?
    |   ++rw groups
    |   |   ++rw fate-sharing-group-size?   uint16
    |   |   ++rw group-color?   string
    |   |   ++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 connection
    |   ++rw encapsulation-type?   identityref
++rw eth-inf-type*   identityref
++rw dot1q-interface
    |   ++rw 12-access-type?   identityref
    |   ++rw dot1q {dot1q}?
    |   |   ++rw physical-inf?   string
    |   |   ++rw c-vlan-id?   uint32
    |   ++rw sub-inf {sub-inf}?
    |   |   ++rw interface-description?
    |   |      |   string
    |   |      ++rw sub-if-id?
    |   |          |   uint32
    |   ++rw qinq {qinq}?
    |   |   ++rw s-vlan-id?   uint32
    |   |   ++rw c-vlan-id?   uint32
    |   ++rw qinany {qinany}?
    |   |   ++rw s-vlan-id?   uint32
    |   |   ++rw atm (atm)?
| ++-rw vpi?         uint32 |
| ++-rw vci?         uint32 |
++-rw vxlan {vxlan}? |
    ++-rw vni-id?       uint32 |
    ++-rw peer-mode?    identityref |
    ++-rw peer-list* [peer-ip] |
        ++-rw peer-ip    inet:ip-address |
++-rw phy-interface |
    ++-rw port-number?  uint32 |
    ++-rw port-speed?   uint32 |
    ++-rw mode?         neg-mode |
    ++-rw phy-mtu?      uint32 |
    ++-rw flow-control? string |
    ++-rw physical-if?  string |
    ++-rw circuit-id?   string |
    ++-rw lldp?         boolean |
    ++-rw oam-802.3ah-link {oam-3ah}? |
        ++-rw enable?    boolean |
        ++-rw uni-loop-prevention? boolean |
++-rw lag-interface {lag-interface}? |
++-rw lag-interface* |
    [lag-interface-number] |
        ++-rw lag-interface-number  uint32 |
++-rw lacp |
    ++-rw lacp-state?   boolean |
    ++-rw lacp-mode?    boolean |
    ++-rw lacp-speed?   boolean |
    ++-rw mini-link?    uint32 |
    ++-rw system-priority? uint16 |
++-rw micro-bfd {micro-bfd}? |
    ++-rw micro-bfd-on-off? |
        enumeration |
    ++-rw bfd-interval?  |
        uint32 |
    ++-rw bfd-hold-timer? |
        uint32 |
++-rw bfd (bfd)? |
    ++-rw bfd-enabled?  |
        boolean |
    ++-rw (holdtime)?  |
        ++: (profile) |
    ++-rw profile-name? |
        string |
    ++-rw (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 link-mtu?
    uint32
++rw oam-802.3ah-link
    (oam-3ah)?
      ++rw enable?    boolean
++rw flow-control?  string
++rw lldp?            boolean
++rw cvlan-id-to-svc-map* [svc-id]
    ++rw svc-id     leafref
    ++rw cvlan-id*  [vid]
    ++rw vid        uint32
++rw split-horizon
    ++rw group-name? string
++rw l2cp-control (L2CP-control)?
    ++rw stp-rstp-mstp?  control-mode
    ++rw pause?          control-mode
    ++rw lACP-lAMP?      control-mode
    ++rw link-oAM?       control-mode
    ++rw esMC?           control-mode
    ++rw l2CP-802.1x?    control-mode
    ++rw e-LMi?          control-mode
    ++rw lldp?           boolean
    ++rw ptp-peer-delay? control-mode
    ++rw garp-mRP?       control-mode
++rw availability
    ++rw access-priority? uint32
++rw (redundancy-mode)?
      :+: (single-active)
      |  ++rw single-active? boolean
      :+: (all-active)
      |  ++rw all-active?  boolean
++rw service
    ++rw svc-input-bandwidth {input-bw}?
      ++rw input-bandwidth* [type]
      |  ++rw type         identityref
      |  ++rw cos-id?      uint8
      |  ++rw vpn-id?      svc-id
      |  ++rw cir?         uint64
      |  ++rw cbs?         uint64
      |  ++rw eir?         uint64
      |  ++rw ebs?         uint64
      |  ++rw pir?         uint64
+-rw pbs?  uint64
+-rw svc-output-bandwidth {output-bw}?
  +-rw output-bandwidth* [type]
    |  +-rw type  identityref
    |  +-rw cos-id?  uint8
    |  +-rw vpn-id?  svc-id
    |  +-rw cir?  uint64
    |  +-rw cbs?  uint64
    |  +-rw eir?  uint64
    |  +-rw ebs?  uint64
    |  +-rw pir?  uint64
    |  +-rw pbs?  uint64
  +-rw qos {qos}?
    +-rw classification-policy
      +-rw rule* [id]
        +-rw id
          |  string
        +-rw (match-type)?
          +-:(match-flow)
            +-rw match-flow
              +-rw dscp?  inet:dscp
              +-rw dot1q?  uint16
              +-rw pcp?  uint8
              +-rw src-mac?  yang:mac-address
              +-rw dst-mac?  yang:mac-address
              +-rw color-type?
                |  identityref
              +-rw target-sites*
                |  svc-id
                |  (target-sites)?
              +-rw any?
                |  empty
              +-rw vpn-id?  svc-id
          +-:(match-application)
            +-rw match-application?
              |  identityref
            +-rw target-class-id?
              |  string
  +-rw qos-profile
    +-rw (qos-profile)?
      +-:(standard)
        +-rw profile?  leafref
++-:(custom)
  ++-rw classes {qos-custom}?
    ++-rw class* [class-id]
      ++-rw class-id
        | string
      ++-rw direction?
        | identityref
      ++-rw policing?
        | identityref
      ++-rw byte-offset?
        | uint16
      ++-rw frame-delay
        +++-rw (flavor)?
          +++-:(lowest)
            | +++-rw use-lowest-latency?
            | empty
          +++-:(boundary)
            | +++-rw delay-bound?
            | uint16
      ++-rw frame-jitter
        +++-rw (flavor)?
          +++-:(lowest)
            | +++-rw use-lowest-jitter?
            | empty
          +++-:(boundary)
            | +++-rw delay-bound?
            | uint32
      ++-rw frame-loss
        ++-rw rate?
          | decimal64
      ++-rw guaranteed-bw-percent
        | decimal64
      ++-rw end-to-end?
        | empty
    ++-rw svc-precedence
      | ++-rw precedence? identityref
    ++-rw broadcast-unknown-unicast-multicast
      ++-rw multicast-site-type?
        | enumeration
      ++-rw multicast-gp-address-mapping* [id]
        | ++-rw id uint16
        | ++-rw vlan-id? uint32
        | ++-rw mac-gp-address?
          | yang:mac-address
        | ++-rw port-lag-number? uint32
      ++-rw bum-overall-rate?
        | uint32
4. Relation with other Yang Models

The L2NM model, aimed at managing the L2VPN Services in a Service Provider Network controller/orchestrator has relations with other Yang modules.
4.1. Relation with L2SM

[RFC8466] defines a L2VPN Service YANG data Model (L2SM) that can be used for communication between customers and VPN service providers. Hence, the model provides inputs to the Network Operator to deliver such service to the customer. Hence, most parts of the model can be directly mapped into L2NM.

- Service requirements: The service requirements can be directly taken from L2SM to L2NM.
- Sites: The sites from L2SM are used to select the Service Provider node. The site information is NOT maintained in L2NM.

4.2. Relation with Network Topology

The L2NM model manages VPN Services running over Service Provider Backbone network. The set of nodes over which it is possible to deploy a L2 VPN Service MAY be part of the topology contained in an ietf-network module.

4.3. Relation with Device Models

Creating services in the l2vpn-ntw module will lead at some point to the configuration of devices. Hence, it is foreseen that the data for the device yang modules will be derived partially from the L2NM vpn-service container. Note that L2NM is NOT a device model.

5. Yang Module

<CODE BEGINS>
module ietf-l2vpn-ntw {
  yang-version 1.1;
  prefix l2vpn-ntw;

  import ietf-inet-types {
    prefix inet;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-netconf-acm {
    prefix nacm;
  }
  organization
    "IETF OPSA (Operations and Management Area) Working Group";
</CODE BEGINS>
This YANG module defines a generic network-oriented model for the management and delivery of Layer 2 VPNs in a Service Provider backbone network.

Copyright (c) 2019 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info).

This version of this YANG module is part of RFC XXXX (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself for full legal notices.

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document are to be interpreted as described in BCP 14 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here.

revision 2019-12-05{
    description
        "Initial revision -00 version";
    reference
        "draft-barguil-opsawg-l2sm-l2nm-00"
        "A Layer 2 VPN Network Yang Model."
};

/* Features */
definition multicast-like{
    description
    "draft-barguil-opsawg-l2sm-l2nm-00"
}
"Enables multicast-like capabilities support in a L2VPN."
}
feature extranet-vpn{
  description
  "Enable the Support of Extranet VPN.";
}
feature target-sites {
  description
  "Enables support of the 'target-sites' match flow parameter.";
}
feature L2CP-control {
  description
  "Enable the Support of L2CP control.";
}
feature input-bw {
  description
  "Enable the support of Input Bandwidth in a VPN.";
}
feature output-bw {
  description
  "Enable the support of Output Bandwidth in a VPN";
}
feature uni-list {
  description
  "Enable the support of UNI list in a VPN.";
}
feature cloud-access {
  description
  "Allow VPN to connect to a Cloud Service provider.";
}
feature oam-3ah {
  description
  "Enables the support of OAM 802.3ah.";
}
feature micro-bfd {
  description
  "Enables the support of Micro-BFD.";
}
feature bfd {
  description
  "Enables the support of BFD."
}
feature signaling-options {
  description
  "Enable the support of signalling option.";
}
feature site-diversity {
    description
    "Enables the support of site diversity constraints in a VPN.";
}
feature encryption {
    description
    "Enables support of encryption.";
}
feature always-on {
    description
    "Enables support for always-on access constraint.";
}
feature requested-type {
    description
    "Enables support for requested-type access constraint.";
}
feature bearer-reference {
    description
    "Enables support for bearer-reference access constraint.";
}
feature qos {
    description
    "Enables support of Class of Services.";
}
feature qos-custom {
    description
    "Enables support of custom qos profile.";
}
feature lag-interface {
    description
    "Enable lag-interface.";
}
feature vlan {
    description
    "Enable the support of VLAN.";
}
feature dot1q {
    description
    "Enable the support of Dot1Q.";
}
feature sub-inf {
    description
    "Enable the support of Sub Interface.";
}
feature qinq {
description
"Enable the support of QinQ."
}
feature qinany {

description
"Enable the support of QinAny."
}
feature atm {

description
"Enable the support of ATM."
}
feature vxlan {

description
"Enable the support of VxLAN."
}
feature ipv4 {

description
"Enables IPv4 support in a VPN."
}
feature ipv6 {

description
"Enables IPv6 support in a VPN."
}
feature lan-tag {

description
"Enables LAN Tag support in a VPN."
}
/* Typedefs */
typedef protocols-type {

type enumeration {
enum "GRE" {
  value 0;
  description "Transport based on GRE."
}
enum "LDP" {
  value 1;
  description "Transport based on LDP."
}
enum "BGP" {
  value 2;
  description "Transport based on BGP."
}
enum "SR" {
  value 3;
  description "Operational status Segment Routing"
}
enum "TE" {
}
typedef operational-type {
  type enumeration {
    enum "up" {
      value 0;
      description "Operational status UP";
    }
    enum "down" {
      value 1;
      description "Operational status DOWN";
    }
    enum "unknown" {
      value 2;
      description "Operational status UNKNOWN";
    }
  }
}
description "This are writable attributes used to identify transport protocols underlaying the service";

typedef operational-type {
  type enumeration {
    enum "up" {
      value 0;
      description "Operational status UP";
    }
    enum "down" {
      value 1;
      description "Operational status DOWN";
    }
    enum "unknown" {
      value 2;
      description "Operational status UNKNOWN";
    }
  }
}
description "This is a read-only attribute used to determine the status of a particular element";

typedef svc-id {
  type string;
description "Defines a type of service component identifier";
}
typedef ccm-priority-type {
  type uint8 {
    range "0..7";
  }
description "A 3 bit priority value to be used in the VLAN tag, if present in the transmitted frame";
}
typedef control-mode {
}
type enumeration {
    enum peer {
        description
        "Peer mode, i.e., participate in the protocol towards the CE.
        Peering is common for LACP and E-LMI and occasionally for LLDP.
        For virtual private services the Subscriber can also request that
        the Service Provider peer spanning tree."
    }
    enum tunnel {
        description
        "Tunnel mode, i.e., pass to the egress or destination site.
        For EPL, the expectation is that L2CP frames are tunneled."
    }
    enum discard {
        description
        "Discard mode, i.e., discard the frame."
    }
}
description
"Defining a type of the control mode on L2CP protocols."
}
typedef neg-mode {
    type enumeration {
        enum full-duplex {
            description
            "Defining Full duplex mode"
        }
        enum auto-neg {
            description
            "Defining Auto negotiation mode"
        }
    }

description
"Defining a type of the negotiation mode"
}

/* Identities */
identity multicast-tree-type {
    description
    "Base identity for multicast tree type."
}
identity ssm-tree-type {
    base multicast-tree-type;
    description
    "Identity for SSM tree type."
}
identity asm-tree-type {
    base multicast-tree-type;
description
"Identity for ASM tree type.";
}

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity one2one{
  base protection-mode;
  description
  "In this scheme, a backup circuit to the primary circuit is provisioned. Depending on the implementation agreement, the protection circuits may either always be in active forwarding state, or may only become active when a faulty state is detected on the primary circuit.";
}

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

identity phy-inf {
  base eth-inf-type;
  description
  "Identity of Physical Interface type";
}

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

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

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

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

identity opaque {
  base bw-type;
description
"Opaque";
}
identity site-type {

description
"Identity of site type.";
}
identity uni {

type site-type;

description
"Identity of User Network Interface ";
}
identity enni {

type site-type;

description
"Identity of External Network to Network Interface";
}
identity service-type {

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

type service-type;

description
"point-to-point Virtual Private Wire Services (VPWS) type.";
}
identity pwe3 {

type service-type;

description
" Pseudo-Wire Emulation Edge to Edge (PWE3) Service type. ";
}
identity ldp-l2tp-vpls {

type service-type;

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

type service-type;

description
"BGP based multipoint Virtual Private LAN services (VPLS) Service Type. This VPLS uses a Border Gateway Protocol (BGP) control plane as described in RFC4761 and RFC6624.";
}
identity vpws-evpn {
base service-type;
description
"VPWS Service Type using Ethernet VPN(EVPN) specified in RFC 7432.";
}
identity pbb-evpn {
base service-type;
description
"PBB Service Type using Ethernet VPN(EVPN) specified in RFC 7432.";
}
identity bundling-type {

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

base bundling-type;
description
"Identity for multiple service bundling,i.e., multiple CE-VLAN IDs can be associated with an L2VPN Service at site.";
}
identity one2one-bundling {

base bundling-type;
description
"Identity for one to one service bundling,i.e., Each L2VPN can be associated with only one CE-VLAN IDs at site.";
}
identity all2one-bundling {

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

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

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

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

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

identity color-type {
  description "Identity of color types";
}

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

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

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

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

identity t-lldp {
base vpn-signaling-type;
description
"Identity of t-lldp.";
}
identity l2tp {
base vpn-signaling-type;
description
"Identity of l2tp.";
}

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

identity vpws-type {
base t-lldp-pwe-type;
description
"Identity for VPWS";
}

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

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

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

identity l2vpn-vpws {
base l2vpn-type;
description
"VPWS L2VPN type.";
}

identity l2vpn-vpls {
base l2vpn-type;
description
"VPLS L2VPN type.";
}

identity distribute-vpls {
base l2vpn-type;
description
"distribute VPLS L2VPN type.";
}

identity evpn-type {
description
"Ethernet VPN types";
}

identity evpn-vpws {
base evpn-type;
description
"VPWS support in EVPN.";
}

identity evpn-pbb {
base evpn-type;
description
"Provider Backbone Bridging Support in EVPN.";
}

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

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

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

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

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

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

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

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

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

identity hub-spoke-disjoint {
  base vpn-topology;
  description "Identity for Hub’n’Spoke VPN topology where Hubs cannot talk between each other.";
}
identity site-role {
  description
  "Base identity for site type.";
}
identity any-to-any-role {
  base site-role;
  description
  "Site in an any to any IPVPN.";
}
identity spoke-role {
  base site-role;
  description
  "Spoke Site in a Hub and Spoke IPVPN.";
}
identity hub-role {
  base site-role;
  description
  "Hub Site in a Hub and 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 frame-delivery-mode {
  description
  "Delivery types"
}

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

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

identity unknown-discard {
  base frame-delivery-mode;
  description
  "Service Frame are conditionally delivered to the destination site and the packet with unknown destination address will be discarded."
identity pw-topo-type {
    description
    "base identity for pw topology Type";
}
identity point-to-point {
    base pw-topo-type;
    description
    "identity for Point to Point.";
}
identity multipoint-to-point {
    base pw-topo-type;
    description
    "identity for Multipoint to point.";
}
identity point-to-multipoint {
    base pw-topo-type;
    description
    "Identity for point to 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 access 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 atm {
base l2-access-type;
description
"ATM Cell.";
}

identity vxlan {
base l2-access-type;
description
"Vxlan access into the vpn";
}
identity provision-model {
  description
  "base identity for provision model.";
}
identity single-side-provision {
  description
  "Identity for single side provisioning with discovery.";
}
identity doubled-side-provision {
  description
  "Identity for double single provisioning.";
}
identity mac-learning-mode {
  description
  "MAC learning mode";
}
identity data-plane {
  base mac-learning-mode;
  description
  "User MAC addresses are learned through ARP broadcast.";
}
identity control-plane {
  base mac-learning-mode;
  description
  "User MAC addresses are advertised through EVPN-BGP";
}
identity vpn-policy-filter-type {
  description
  "Base identity for filter type.";
}
identity lan {
  base vpn-policy-filter-type;
  description
  "Identity for lan tag filter type.";
}
identity mac-action {
  description
  "Base identity for MAC action.";
}
identity drop {
  base mac-action;
  description
  "Identity for packet drop.";
}
identity flood {
  base mac-action;
  description
  "Identity for packet flooding.";
}

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

identity load-balance-method {
  description
  "Base identity for load balance method.";
}

identity fat-pw {
  base load-balance-method;
  description
  "Identity for Fat PW. Fat label is applied to Pseudowires across MPLS network.";
}

identity entropy-label {
  base load-balance-method;
  description
  "Identity for entropy label. Entropy label is applied to IP forwarding, L2VPN or L3VPN across MPLS network";
}

identity vxlan-source-port {
  base load-balance-method;
  description
  "Identity for vxlan source port. VxLAN Source Port is one load balancing method.";
}

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity precedence-type {
    description
    "Redundancy type. The service can be created
    with active and backup signalization.";
}

identity primary {
    base precedence-type;
    description
    "Identifies the Main L2VPN.";
}

identity backup {
    base precedence-type;
    description
    "Identifies the Backup L2VPN.";
}

/* Groupings */

grouping svc-transport-encapsulation {
    container transport-encapsulation {
        leaf protocol {
            type protocols-type;
            description
            "Protocols used to support transport";
        }
        description
        "";
    }
    description
    "";
}

/* Groupings */

grouping split-horizon-group {
    container split-horizon {
        leaf group-name {
            type string;
            description
            "group-name of the Split Horizon";
        }
        description
        "Configuration with split horizon enabled";
    }
    description
}
"Configuration with split horizon enabled";
}

grouping status-params {
    container status {
        leaf admin-enabled {
            type boolean;
            description
            "Administrative Status UP/DOWN";
        }
        leaf oper-status {
            type operational-type;
            config false;
            description
            "Operations status";
        }
        description "";
    }
    description
    "Grouping used to join operational and administrative status
    is re used in the Site Network Acess and in the VPN-Node";
}

grouping site-bearer-params {
    container site-bearers {
        leaf bearer-id {
            type string;
            description
            "Unique identifier for a bearer. This
            identifies shall be mapped to the bearer-reference
            on a vpn-network-access.";
        }
        description
        "Container that encloses all the bearers
        connected to a site. A bearer is mapped one to one
        to a port on the PE router.";
    }
    description
    "Container that encloses all the bearers
    connected to a site. A bearer is mapped one to one
    to a port on the PE router.";
}

grouping svc-precedence {
    container svc-precedence {
        leaf precedence{
            type identityref {
                base precedence-type;
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-ntw/sites/site/site-id";
            }
            description "Site ID to be authorized.";
          }
        }
        case permit-any-except {
          leaf-list deny-site {
            type leafref {
              path "/l2vpn-ntw/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-ntw/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-ntw/sites/site/site-id";
            }
            description
                "Site ID."
        }
        description
            "List of denied sites."
    }
    description
        "Configuration of denied sites."
}

description
    "Cloud access configuration."
}

description
"Container for cloud access configurations";
}
description
"Grouping for vpn cloud definition";
}
grouping site-device {
container device {
list devices {
key "device-id";
leaf device-id {
  type string;
  description
    "Device ID";
}
}
leaf location {
  type leafref {
    path "/l2vpn-ntw/sites/site/locations/location/location-id";
  }
  description
    "Site name";
}
container management {
  leaf address {
    type inet:ip-address;
    description
      "Address";
  }
  leaf management-transport {
    type identityref {
      base address-family;
    }
    description
      "Transport protocol used for management.";
  }
  description
    "Container for management";
}
description
"List of devices";
}
description
"Devices configuration";
}
description
"Device parameters for the site.";
grouping site-management {
  container management {
    leaf type {
      type identityref {
        base management;
      } 
      description
      "Management type of the connection.";
    }
    description
    "Container for management";
  }
  description
  "Grouping for management";
}

grouping site-vpn-policy {
  container vpn-policies {
    list vpn-policy {
      key vpn-policy-id;
      leaf vpn-policy-id {
        type string;
        description
        "Unique identifier for the VPN policy.";
      }
      list entries {
        key id;
        leaf id {
          type string;
          description
          "Unique identifier for the policy entry.";
        }
      }
      container filters {
        list filter {
          key type;
          ordered-by user;
          leaf type {
            type identityref {
              base vpn-policy-filter-type;
            } 
            description
            "Type of VPN Policy filter.";
          }
          leaf-list lan-tag {
            when "derived-from-or-self(../type, 'l2vpn-ntw:lan')" {
          }
        }
      }
    }
  }
}

description
"Only applies when VPN Policy filter is LAN Tag filter."
}
if-feature lan-tag;
type uint32;
description
"List of Ethernet LAN Tag to be matched. Ethernet LAN Tag identifies a particular broadcast domain in a VPN. ";

/*leaf-list ipv4-lan-prefix {
when "derived-from-or-self(../type, 'l2vpn-ntw:ipv4')" {
description
"Only applies when VPN Policy filter is IPv4 Prefix filter."
}
if-feature ipv4;
type inet:ipv4-prefix;
description
"List of IPv4 prefixes as LAN Prefixes to be matched.";
}"*/
/*leaf-list ipv6-lan-prefix {
when "derived-from-or-self(../type, 'l2vpn-ntw:ipv6')" {
description
"Only applies when VPN Policy filter is IPv6 Prefix filter."
}
if-feature ipv6;
type inet:ipv6-prefix;
description
"List of IPv6 prefixes as LAN prefixes to be matched.";
}*/
description
"List of filters used on the site. This list can be augmented.";

description
"If a more-granular VPN attachment is necessary, filtering can be used. If used, it permits the splitting of site LANs among multiple VPNs. The Site LAN can be split based on either LAN-tag or LAN prefix. If no filter is used, all the LANs will be part of the same VPNs with the same role.";

list vpn {
key vpn-id;
leaf vpn-id {
type leafref {
path "/l2vpn-ntw/vpn-services/"+
"vpn-svc/vpn-id";
}
mandatory true;
description "Reference to an IP VPN."
}
leaf site-role {
type identityref {
base site-role;
}
default any-to-any-role;
description "Role of the site in the IP VPN."
}
description "List of VPNs the LAN is associated with."
}
description "List of entries for export policy."
}
description "List of VPN policies."
}
description "VPN policy parameters for the site."
}

grouping bum-frame-delivery {
container bum-frame-delivery-modes {
list bum-frame-delivery {
key traffic-type;
leaf traffic-type {
type identityref {
base tf-type;
}
description "Type of frame delivery. It support unicast frame delivery, multicast frame delivery and broadcast frame delivery."
}
leaf delivery-mode {
type identityref {
base frame-delivery-mode;
}
description "Define Frame Delivery Mode (unconditional[default], conditional, or discard)."
}
grouping cvlan-svc-map-grouping {
  list cvlan-id-to-svc-map {
    key "svc-id";
    leaf svc-id {
      type leafref {
        path "/l2vpn-ntw/vpn-services/vpn-svc/vpn-id";
      }
      description "VPN Service identifier";
    }
  list cvlan-id {
    key vid;
    leaf vid {
      type uint32;
      description "CVLAN ID";
    }
    description "List of CVLAN-ID to SVC Map configurations";
  }
  list cvlan-id-to-L2VPN-svc-map {
    key location-id;
    leaf location-id {
      type string; 
      description "Location ID";
    }
    leaf address {
      type string;
    }
  }
}

description
 "Address (number and street) of the site.";
} 
leaf zip-code {
 type string;
 description
 "ZIP code of the site.";
}
leaf state {
 type string;
 description
 "State of the site. This leaf can also be used to
 describe a region for country who does not have
 states.";
}
leaf city {
 type string;
 description
 "City of the site.";
}
leaf country-code {
 type string;
 description
 "Country of the site.";
} 
description
 "List for location";
} 
description
 "Location of the site.";
} 
description
 "This grouping defines customer location parameters";
} 
grouping site-diversity {
 container site-diversity {
 if-feature site-diversity;
 container groups {
 list group {
 key group-id;
 leaf group-id {
 type string;
 description
 "Group-id the site is belonging to";
 } 
 description


"List of group-id";
}
description
"Groups the site is belonging to.
All site network accesses will inherit those group values.";
}
description
"Diversity constraint type.";
}
description
"This grouping defines site diversity parameters";
}
grouping site-service {

description
"This grouping defines site service parameters";
}
grouping vpn-service-multicast {
container multicast-like {
if-feature multicast-like;
leaf enabled {

type boolean;

default false;

description
"Enables multicast.";
}
container customer-tree-flavors {
leaf-list tree-flavor {

type identityref {

base multicast-tree-type;
}
description
"Type of tree to be used.";
}
description
"Type of trees used by customer.";

leaf traffic-type {

type identityref {

base tf-type;
}
description
"Traffic Type";

*/
uses bum-frame-delivery;
leaf multicast-gp-port-mapping {

}
type identityref {
    base mapping-type;
}
description "Describe the way in which each interface is associated with the Multicast group";

description "Multicast global parameters for the VPN service.";

description "Grouping for multicast VPN definition."

grouping vpn-extranet {
    container extranet-vpns {
        if-feature extranet-vpn;
        list extranet-vpn {
            key vpn-id;

            leaf vpn-id {
                type svc-id;
                description "Identifies the target VPN.";
            }

            leaf local-sites-role {
                type identityref {
                    base site-role;
                }

                default any-to-any-role;

                description "This describes the role of the local sites in the target VPN topology.";
            }

            description "List of extranet VPNs the local VPN is attached to.";
        }

        description "Container for extranet VPN configuration.

        Grouping for extranet VPN configuration. This provides an easy way to interconnect all sites from two VPNs.";
    }

    grouping signaling-options-grouping {
        list signaling-options {
            key "type";

            leaf type {
        

type identityref {
  base vpn-signaling-type;
}
description
"VPN signaling types";
}
container l2vpn-bgp {
when "./type = 'l2vpn-bgp'" {
  description
  "Only applies when vpn signaling type is l2vpn BGP protocol.";
}
leaf vpn-id {
  type leafref{
      path "/l2vpn-ntw/vpn-services/vpn-svc/vpn-id";
  }
  description
  "Identifies the target VPN";
}
leaf type {
  type identityref {
    base l2vpn-type;
  }
  description
  "L2VPN types";
}
leaf pwe-encapsulation-type {
  type identityref {
    base pwe-encapsulation-type;
  }
  description
  "PWE Encapsulation Type";
}
container pwe-mtu {
  leaf allow-mtu-mismatch {
      type boolean;
      description
      "Allow MTU mismatch";
  }
  description
  "Container of PWE MTU configurations";
}
leaf address-family {
  type identityref {
    base address-family;
  }
  description
  "Address family used for management.";
}
description
"Container for MP BGP L2VPN";
}
container evpn-bgp {
when "../type = 'evpn-bgp'" {

description
"Only applies when vpn signaling type is EVPN BGP protocol.";
}
leaf vpn-id {
  type leafref{
    path "/l2vpn-ntw/vpn-services/vpn-svc/vpn-id";
  }

description
"Identifies the target EVPN";
}
leaf type {
  type identityref {
    base evpn-type;
  }

description
"L2VPN types";
}
leaf address-family {
  type identityref {
    base address-family;
  }

description
"Address family used for management.";
}
leaf mac-learning-mode {
  type identityref {
    base mac-learning-mode;
  }

description
"Indicates through which plane MAC addresses are advertised.";
}
leaf arp-suppress {
  type boolean;
  default false;

description
"Indicates whether to suppress ARP broadcast.";
}

description
"Container for MP BGP L2VPN";
}
container t-ldp-pwe {

when "./type = 't-ldp'" {
  description
  "Only applies when vpn signaling type is Target LDP.";
}
leaf type {
  type identityref {
    base t-ldp-pwe-type;
  }
  description
  "T-LDP PWE type";
}
leaf pwe-encapsulation-type {
  type identityref {
    base pwe-encapsulation-type;
  }
  description
  "PWE Encapsulation Type.";
}
leaf control-word {
  type boolean;
  description
    "Control word configurations";
}
container pwe-mtu {
  leaf allow-mtu-mismatch {
    type boolean;
    description
      "Allow MTU mismatch";
  }
  description
    "Container of PWE MTU configurations";
}
leaf provision-mode {
  type identityref {
    base provision-model;
  }
  description
    "provision-models. It support Double-Sided Provisioning
    model or single-side provisioning model.";
}
list ac-pw-list {
  key "peer-addr vc-id";
  leaf peer-addr {
    type inet:ip-address;
    description
      "Peer IP address.";
  }
  leaf vc-id {
type string;
description
"VC label used to identify PW."

leaf ac-group-id {
type leafref{
path "../../../../../vpn-network-accesses/vpn-network-access/"+
"access-diversity/groups/group/group-id";
} 
description
"Reference to Network Access Group ID.";
}/*
leaf pw-type {
type identityref {
base pw-topo-type;
} 
description
"PW topology type";
}
leaf pw-priority {
type uint32;
description
"Defines the priority for the PW.
The higher the pw-priority value,
the higher the preference of the PW will be.";
} 
description
"List of AC and PW bindings.";
}
container qinq {
when "./.type = 'h-vpls'" {
description
"Only applies when t-ldp pwe type is h-vpls."
}
leaf s-tag {
type uint32;
description
"S-TAG";
}
leaf c-tag {
type uint32;
description
"C-TAG";
} 
description
"Container for QinQ";
}
"Container of T-LDP PWE configurations";
}
container l2tp-pwe {
when "./.type = 'l2tp'" {
description
"Only applies when vpn signaling type is L2TP protocol.";
}
leaf type {
    type identityref {
        base t-ldp-pwe-type;
    }
    description
    "T-LDP PWE type";
}
leaf encapsulation-type {
    type identityref {
        base encapsulation-type;
    }
    description
    "Encapsulation type";
}
leaf control-word {
    type boolean;
    description
    "Control word configurations";
}
list ac-pw-list {
    key "peer-addr vc-id";
    leaf peer-addr {
        type inet:ip-address;
        description
        "Peer IP address.";
    }
    leaf vc-id {
        type string;
        description
        "VC lable used to identify PW.";
    }
    /*
    leaf ac-group-id {
    type leafref{
        path "/./././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././././ ./
leaf pw-type {
  type identityref {
    base pw-topo-type;
  }
  description "PW topology type";
}
leaf pw-priority {
  type uint32;
  description "PW priority";
}

description "List of AC and PW bindings.";
}
description "Container for l2tp pw";
}
description "List of VPN Signaling Option.";
}
description "Grouping for signaling option";
}
/*
grouping load-balance-grouping {
  leaf enable {
    type boolean;
    description "Enable load balancing";
  }
  leaf load-balance-method {
    type identityref {
      base load-balance-method;
    }
    description "select load balancing method such as fat-pw, entropy-label, or vxlan-source-udp-port.";
  }
  description "Grouping for load balance ";
}
*/
grouping operational-requirements-ops {
  leaf actual-site-start {


type yang:date-and-time;
config false;
description
"Optional leaf indicating actual date
and time when the service at a particular
site actually started";
leaf actual-site-stop {
  type yang:date-and-time;
  config false;
description
  "Optional leaf indicating actual date
  and time when the service at a particular
  site actually stopped";
}

leaf bundling-type {
  type identityref {
    base bundling-type;
  }
description
  "Bundling type";
}

leaf default-ce-vlan-id {
  type uint32;
description
  "Default CE VLAN ID set at site level.";
}

description
"This grouping defines some operational parameters parameters";

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";
}
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";
    }
    description
    "Grouping for y.1731";
}

/*
grouping site-security {
    container security {
        uses mac-loop-prevention-grouping;
        container access-control-list {
            list mac {
                key "mac-address";
                leaf mac-address {
                    type yang:mac-address;
                    description
                    "MAC address";
                }
                description
                "List for MAC";
            }
            description
            "Container for access control";
        }
        uses mac-addr-limit-grouping;
    }
*/
uses cvlan-svc-map-grouping;
description
"Security parameters";
}*

grouping acl-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 List.";
}*

grouping lacp-grouping {
    container lacp {
        leaf lacp-state {
            type boolean;
            description
"LACP on/off";
        }
        leaf lacp-mode {
            type boolean;
            description
"LACP mode";
        }
        leaf lacp-speed {
            type boolean;
            description
"LACP speed";
        }
        leaf mini-link {
            type uint32;
            description
"The minimum aggregate bandwidth for a LAG";
    }
leaf system-priority {
  type uint16;
  description
    "Indicates the LACP priority for the system.
    The range is from 0 to 65535.
    The default is 32768.";
}
container micro-bfd {
  if-feature micro-bfd;
  leaf micro-bfd-on-off {
    type enumeration {
      enum on {
        description
          "Micro-bfd on";
      }
      enum off {
        description
          "Micro-bfd off";
      }
    }
    description
      "Micro BFD ON/OFF";
  }
  leaf bfd-interval {
    type uint32;
    description
      "BFD interval";
  }
  leaf bfd-hold-timer {
    type uint32;
    description
      "BFD hold timer";
  }
  description
    "Container of Micro-BFD configurations";
}
container bfd {
  if-feature bfd;
  leaf bfd-enabled {
    type boolean;
    description
      "BFD activation";
  }
  choice holdtime {
    case profile {
      leaf profile-name {
        type string;
      }
    }
  }
}
description
"Service provider well known profile."
}

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 link-mtu {
      type uint32;
      description
      "Link MTU size.";
    }
    container oam-802.3ah-link {
      if-feature oam-3ah;
      leaf enable {
        type boolean;
      }
    }
  }
}
description
"Indicate whether support oam 802.3 ah link";
}
description
"Container for oam 802.3 ah link.";
}
description
"Member link";
}
description
"Container of Member link list";
}
leaf flow-control {
type string;
description
"Flow control";
}
leaf lldp {
type boolean;
description
"LLDP";
}
description
"LACP";
}
description
"Grouping for lACP";
}
grouping phy-interface-grouping {
container phy-interface {
leaf port-number {
type uint32;
description
"Port number";
}
leaf port-speed {
type uint32;
description
"Port speed";
}
leaf mode {
type neg-mode;
description
"Negotiation mode";
}
leaf phy-mtu {
  type uint32;
  description
    "PHY MTU";
}

leaf flow-control {
  type string;
  description
    "Flow control";
}

leaf physical-if {
  type string;
  description
    "Physical interface";
}

leaf circuit-id {
  type string;
  description
    "Circuit ID";
}

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

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

leaf uni-loop-prevention {
  type boolean;
  description
    "If this leaf set to truth that the port automatically
     goes down when a physical loopback is detect.";
}

description
    "Container of PHY Interface Attributes configurations";
}
description
"Grouping for phy interface."
}
grouping lag-interface-grouping {
container lag-interface {
if-feature lag-interface;
list lag-interface {
key "lag-interface-number";
leaf lag-interface-number {
type uint32;
description
"LAG interface number";
}
uses lacp-grouping;
description
"List of LAG interfaces";
}
description
"Container of LAG interface attributes configuration";
}
description
"Grouping for LAG interface";
}
grouping dot1q-interface-grouping {
container dot1q-interface {
leaf l2-access-type {
type identityref {
base l2-access-type;
}
description
"L2 Access Encapsulation Type";
}
container dot1q {
when "/l2-access-type='dot1q'";
if-feature dot1q;
leaf physical-inf {
type string;
description
"Physical Interface";
}
leaf c-vlan-id {
type uint32;
description
"VLAN identifier";
}
description
"Qot1q";
container sub-inf {
  when ""./l2-access-type='sub-interface'";
  if-feature sub-inf;
  leaf interface-description {
    type string;
    description
    "Interface description";
  }
  leaf sub-if-id {
    type uint32;
    description
    "Sub interface ID";
  }
  description
  "sub interface";
}

container qinq {
  when ""./l2-access-type='qinq'";
  if-feature qinq;
  leaf s-vlan-id {
    type uint32;
    description
    "S-VLAN Identifier";
  }
  leaf c-vlan-id {
    type uint32;
    description
    "C-VLAN Identifier";
  }
  description
  "QinQ";
}

container qinany {
  if-feature qinany;
  leaf s-vlan-id {
    type uint32;
    description
    "S-Vlan ID";
  }
  description
  "Container for Q in Any";
}

container atm {
  when ""./l2-access-type='atm'";
  if-feature atm;
  leaf vpi {
    type uint32;
    description
    "VPI";
  }
  leaf vci {
    type uint32;
    description
    "VCI";
  }
  description
  "ATM";
}
type uint32;
description
"VPI Identifier";
}
leaf vci {
type uint32;
description
"VCI Identifier";
}
description
"ATM Cell.";
}
container vxlan {
when "/l2-access-type='vxlan'";
if-feature vxlan;
leaf vni-id {
type uint32;
description
"VNI Identifier";
}
leaf peer-mode {
type identityref {
base vxlan-peer-mode;

description
"specify the vxlan access mode";
}
list peer-list {
key peer-ip;
leaf peer-ip {

type inet:ip-address;
description
"Peer IP";
}
description
"List for peer IP";
}
description
"QinQ";

description
"Container for dot1Q Interface";
}
description
"Grouping for Layer2 access";
}

grouping ethernet-connection-grouping {
container connection {
leaf encapsulation-type {
type identityref {
base encapsulation-type;
}
description
"Encapsulation Type";
}
leaf-list eth-inf-type {
type identityref {
base eth-inf-type;
}
description
"Ethernet Interface Type";
}
uses dot1q-interface-grouping;
uses phy-interface-grouping;
uses lag-interface-grouping;
uses cvlan-svc-map-grouping;
uses split-horizon-group;
uses l2cp-grouping;
description
"Container for bearer";
}
description
"Grouping for bearer.";
}

grouping svc-mtu-grouping {
leaf svc-mtu {
type uint32;
description
"SVC MTU, it is also known as the maximum transmission unit or
maximum frame size, When a frame is larger than the MTU, it is
broken down, or fragmented, into smaller pieces by the network protocol
to accommodate the MTU of the network";
}
description
"Grouping for service mtu.";
}

grouping svc-preservation-grouping {
leaf ce-vlan-preservation {
type boolean;
description
"Preserve the CE-VLAN ID from ingress to egress, i.e.,
CE-VLAN tag of the egress frame are identical to
those of the ingress frame that yielded this
egress service frame. If All-to-One bundling within a site is Enabled, then preservation applies to all Ingress service frames. If All-to-One bundling is Disabled, then preservation applies to tagged Ingress service frames having CE-VLAN ID 1 through 4094.

leaf ce-vlan-cos-perservation {
  type boolean;
  description
  "CE VLAN CoS preservation. PCP bits in the CE-VLAN tag of the egress frame are identical to those of the ingress frame that yielded this egress service frame."
};

grouping mac-addr-limit-grouping {
  container mac-addr-limit {
    leaf mac-num-limit {
      type uint16;
      description
      "maximum number of MAC addresses learned from the subscriber for a single service instance."
    }
    leaf time-interval {
      type uint32;
      units milliseconds;
      description
      "The aging time of the mac address."
    }
    leaf action {
      type identityref {
        base mac-action;
      }
      description
      "specify the action when the upper limit is exceeded: drop the packet, flood the packet, or simply send a warning log message."
    }
  }
  description
  "Grouping for mac address limit"
};
grouping availability-grouping {
  container availability {
    leaf access-priority {
      type uint32;
      description "Access priority";
    }
    choice redundancy-mode {
      case single-active {
        leaf single-active {
          type boolean;
          description "Single active";
        }
        description "Single active case";
      }
      case all-active {
        leaf all-active {
          type boolean;
          description "All active";
        }
        description "All active case";
      }
      description "Redundancy mode choice";
    }
    description "Container of availability optional configurations";
  }
  description "Grouping for availability";
}

grouping l2cp-grouping {
  container l2cp-control {
    if-feature L2CP-control;
    leaf stp-rstp-mstp {
      type control-mode;
      description "STP/RSTP/MSTP protocol type applicable to all UNIs";
    }
    leaf pause {
      type control-mode;
      description "Pause protocol type applicable to all UNIs";
    }
  }
}
leaf lACP-LAMP {
    type control-mode;
    description "LACP/LAMP";
}
leaf link-oam {
    type control-mode;
    description "Link OAM";
}
leaf esmc {
    type control-mode;
    description "ESMC";
}
leaf l2cp-802.1x {
    type control-mode;
    description "802.x";
}
leaf e-lmi {
    type control-mode;
    description "E-LMI";
}
leaf lldp {
    type boolean;
    description "LLDP protocol type applicable to all UNIs";
}
leaf ptp-peer-delay {
    type control-mode;
    description "PTP peer delay";
}
leaf garp-mrp {
    type control-mode;
    description "GARP/MRP";
}
description "Container of L2CP control configurations";
description "Grouping for l2cp control";
grouping B-U-M-grouping {
  container broadcast-unknown-unicast-multicast {
    leaf multicast-site-type {
      type enumeration {
        enum receiver-only {
          description "The site only has receivers.";
        }
        enum source-only {
          description "The site only has sources.";
        }
        enum source-receiver {
          description "The site has both sources and receivers.";
        }
        default "source-receiver";
      }
    }
  }
  list multicast-gp-address-mapping {
    key id;
    leaf id {
      type uint16;
      description "Unique identifier for the mapping.";
    }
    leaf vlan-id {
      type uint32;
      description "the VLAN ID of the Multicast group";
    }
    leaf mac-gp-address {
      type yang:mac-address;
      description "the MAC address of the Multicast group";
    }
    leaf port-lag-number {
      type uint32;
      description "the ports/LAGs belonging to the Multicast group";
    }
    description "List of Port to group mappings.";
  }
}
leaf bum-overall-rate {
  type uint32;
  description
    "overall rate for BUM";
}
list bum-rate-per-type {
  key "type";
  leaf type {
    type identityref {
      base bum-type;
    }
    description
      "BUM type";
  }
  leaf rate {
    type uint32;
    description
      "rate for BUM";
  }
  description
    "List of rate per type";
}
description
  "Container of broadcast, unknown unicast, and multicast configurations";
}
description
  "Grouping for broadcast, unknown unicast, and multicast ";
}
grouping mac-loop-prevention-grouping {
  container mac-loop-prevention {
    leaf frequency {
      type uint32;
      description
        "Frequency";
    }
    leaf protection-type {
      type identityref {
        base loop-prevention-type;
      }
      description
        "Protection type";
    }
    leaf number-retries {
      type uint32;
      description
        "Number of retries";
    }
  }
}

description
"Container of MAC loop prevention.";
}
description
"Grouping for MAC loop prevention";
}
grouping ethernet-svc-oam-grouping {
    container ethernet-service-oam {
        leaf md-name {
            type string;
            description
            "Maintenance domain name";
        }
        leaf md-level {
            type uint8;
            description
            "Maintenance domain level";
        }
        container cfm-802.1-ag {
            list n2-uni-c {
                key "maid";
                uses cfm-802-grouping;
                description
                "List of UNI-N to UNI-C";
            }
            list n2-uni-n {
                key "maid";
                uses cfm-802-grouping;
                description
                "List of UNI-N to UNI-N";
            }
            description
            "Container of 802.1ag CFM configurations.";
        }
        uses y-1731;
        description
        "Container for Ethernet service OAM.";
    }
    description
    "Grouping for Ethernet service OAM.";
}
grouping fate-sharing-group {
    container groups {
        leaf fate-sharing-group-size {
            type uint16;
        }
    }
}
description
"Fate sharing group size.";
}
leaf group-color {
    type string;
    description
"Group color associated with a particular VPN.";
}
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 vpn-network-access
is belonging to.";
    }
    description
"Grouping definition to assign
group-ids to site or vpn-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, ATM, Frame
            Relay, IP Layer 2 Transport, Frame Relay DLCI,
            SONET/SDH,PPP.";
        }
        leaf strict {
            type boolean;
            default false;
            description
            "Define if the requested-type is a preference
            or a strict requirement.";
        }
        description
            "Container for requested type.";
    }
    leaf always-on {
        if-feature always-on;
        type boolean;
        default true;
        description
            "Request for an always on access type.
            This means no Dial access type for"
leaf bearer-reference {
  if-feature bearer-reference;
  type string;
  description
    "This is an internal reference for the
     service provider.";
}
description
  "Bearer specific parameters.
   To be augmented.";
}
description
  "Grouping to define physical properties of
   a site attachment.";
}
grouping vpn-attachment-grouping {
  container vpn-attachment {
    leaf device-id {
      type string;
      description
        "Device ID";
    }
  }
  container management {
    leaf address-family {
      type identityref {
        base address-family;
      }
      description
        "Address family used for management.";
    }
    leaf address {
      type inet:ip-address;
      description
        "Management address";
    }
    description
      "Management configuration..";
  }
  choice attachment-flavor {
    case vpn-flavor {
      list vpn-flavor {
        key vpn-id;
        leaf vpn-id {
          type leafref {

path "/l2vpn-ntw/vpn-services"+
  "/vpn-svc/vpn-id";
} description
"Reference to a VPN.";
} leaf site-role {
  type identityref {
    base site-role;
  }
  default any-to-any-role;
  description
  "Role of the site in the IPVPN.";
} description
"List of IPVPNs attached by the Site Network Access";
}
} case vpn-policy-id {
  leaf vpn-policy-id {
    type leafref {
      path "/l2vpn-ntw/sites/site/vpn-policies/vpn-policy/vpn-policy-id";
    } description
    "Reference to a vpn policy";
  }
  mandatory true;
  description
    "Choice for VPN attachment flavor.";
} description
"Defines VPN attachment of a site.";
} description
"Grouping for access attachment";
}
grouping site-service-basic {
  container svc-input-bandwidth {
    if-feature input-bw;
    list input-bandwidth {
      key "type";
      leaf type {
        type identityref {
          base bw-type;
        }
        description
        "";
      }
    }
  }
}

"Bandwidth Type";
}
leaf cos-id {
  type uint8;
  description
  "Identifier of Class of Service
    , indicated by DSCP or a CE-CLAN
    CoS(802.1p)value in the service frame.";
}
leaf vpn-id {
  type svc-id;
  description
  "Identifies the target VPN.";
}
leaf cir {
  type uint64;
  description
  "Committed Information Rate. The maximum number of
    bits that a port can receive or send during
    one-second over an interface.";
}
leaf cbs {
  type uint64;
  description
  "Committed Burst Size. CBS controls the bursty nature
    of the traffic. Traffic that does not use the configured
    CIR accumulates credits until the credits reach the
    configured CBS.";
}
leaf eir {
  type uint64;
  description
  "Excess Information Rate, i.e., Excess frame delivery
    allowed not subject to SLA. The traffic rate can be
    limited by eir.";
}
leaf ebs {
  type uint64;
  description
  "Excess Burst Size. The bandwidth available for burst
    traffic from the EBS is subject to the amount of bandwidth
    that is accumulated during periods when traffic allocated
    by the EIR policy is not used.";
}
leaf pir{
  type uint64;
  description
"Peak Information Rate, i.e., maximum frame delivery allowed. It is equal to or less than sum of cir and eir."
}
leaf pbs {
    type uint64;
    description
    "Peak Burst Size. It is measured in bytes per second.";
}
description
"List for input bandwidth";
}
description
"From the PE perspective, the service input bandwidth of the connection."
}
container svc-output-bandwidth {
    if-feature output-bw;
    list output-bandwidth {
        key "type";
        leaf type {
            type identityref {
                base bw-type;
            }
            description
            "Bandwidth Type";
        }
        leaf cos-id {
            type uint8;
            description
            "Identifier of Class of Service, indicated by DSCP or a CE-CLAN CoS(802.1p)value in the service frame.";
        }
        leaf vpn-id {
            type svc-id;
            description
            "Identifies the target VPN.";
        }
        leaf cir {
            type uint64;
            description
            "Committed Information Rate. The maximum number of bits that a port can receive or send during one-second over an interface.";
        }
        leaf cbs {
            type uint64;
            description
            "Peak Burst Size. It is measured in bytes per second.";
        }
    }
}

"Committed Burst Size. CBS controls the bursty nature of the traffic. Traffic that does not use the configured CIR accumulates credits until the credits reach the configured CBS."

leaf eir {
  type uint64;
  description
  "Excess Information Rate, i.e., Excess frame delivery allowed not subject to SLA. The traffic rate can be limited by eir."
}

leaf ebs {
  type uint64;
  description
  "Excess Burst Size. The bandwidth available for burst traffic from the EBS is subject to the amount of bandwidth that is accumulated during periods when traffic allocated by the EIR policy is not used."
}

leaf pir {
  type uint64;
  description
  "Peak Information Rate, i.e., maximum frame delivery allowed. It is equal to or less than sum of cir and eir."
}

leaf pbs {
  type uint64;
  description
  "Peak Burst Size. It is measured in bytes per second."
}

description
  "List for output bandwidth"

description
  "From the PE perspective, the service output bandwidth of the connection."

description
  "Grouping for site service"

grouping flow-definition {
  container match-flow {
    leaf dscp {
      type inet:dscp;
      description
      "DSCP value."
    }
  }
}
leaf dot1q {
    type uint32;
    description
        "802.1q matching. It is VLAN Tag added into frame.";
}
leaf pcp {
    type uint8{
        range "0 .. 7";
    }
    description
        "PCP value";
}
leaf src-mac {
    type yang:mac-address;
    description
        "Source MAC";
}
leaf dst-mac {
    type yang:mac-address;
    description
        "Destination MAC";
}
leaf color-type {
    type identityref {
        base color-type;
    }
    description
        "Color Types";
}
leaf-list target-sites {
    type svc-id;
    description
        "Identify a site as traffic destination.";
}
leaf any {
    type empty;
    description
        "Allow all.";
}
leaf vpn-id {
    type svc-id;
    description
        "Reference to the target VPN.";
}

}
description
"Flow definition based on criteria."
}
grouping services-grouping {
container service {
uses site-service-qos-profile;
description
"Container for service";
}
description
"Grouping for Services";
}

grouping service-grouping {
container service {
uses site-service-basic;
uses site-service-qos-profile;
description
"Container for service";
}
description
"Grouping for service.";
}

grouping site-service-qos-profile {
container qos {
if-feature qos;
container classification-policy {
list rule {
key "id";
ordered-by user;
leaf id {
type string;
description
"A description identifying qos classification policy rule.";
}
choice match-type {
default "match-flow";
case match-flow {
container match-flow {
leaf dscp {
type inet:dscp;
description
"DSCP value.";
}
leaf dot1q {
    type uint16;
    description
    "802.1q matching. It is VLAN Tag added into frame.";
}
leaf pcp {
    type uint8 {
        range "0 .. 7";
    }
    description
    "PCP value.";
}
leaf src-mac {
    type yang:mac-address;
    description
    "Source MAC";
}
leaf dst-mac {
    type yang:mac-address;
    description
    "Destination MAC";
}
leaf color-type {
    type identityref {
        base color-type;
    }
    description
    "Color Types.";
}
leaf-list target-sites {
    if-feature target-sites;
    type svc-id;
    description
    "Identify a site as traffic destination.";
}
leaf any {
    type empty;
    description
    "Allow all.";
}
leaf vpn-id {
    type svc-id;
    description
    "Reference to the target VPN.";
}

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

container classes {
  if-feature qos-custom;
  list class {
    key "class-id";
    leaf class-id {
      type string;
      description "Identification of the class of service. This identifier is internal to the administration.";
    }
    leaf direction {
      type identityref {
        base qos-profile-direction;
      }
      default "bidirectional";
      description "The direction which QoS profile is applied to. By default, the direction is bidirectional.";
    }
    leaf policing {
      type identityref {
        base policing;
      }
      default "one-rate-two-color";
      description "The policing can be either one-rate, two-color (1R2C) or two-rate, three-color (2R3C). By default, the policing is on rate two color.";
    }
    leaf byte-offset {
      type uint16;
      description "For not including extra VLAN tags in the QoS calculation.";
    }
  }
  container frame-delay {
    choice flavor {
      case lowest {
        leaf use-lowest-latency {
          type empty;
          description "The traffic class should use the lowest delay path.";
        }
      }
    }
  }
}
case boundary {
  leaf delay-bound {
    type uint16;
    units "msec";
    description
    "The traffic class should use
    a path with a defined maximum
    delay.";
  }
}

description
"Delay constraint on the traffic
class.";

description
"Delay constraint on the traffic
class.";

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

description
"Jitter constraint on the traffic
class.";

description
"Jitter constraint on the traffic
class.";

container frame-loss {
leaf rate {
    type decimal64 {
        fraction-digits 2;
        range "0..100";
    }
    units "percent";
    description
        "Frame Loss rate constraint on the traffic class.";
}

description
    "Container for frame loss rate.";
}

container bandwidth {
    leaf guaranteed-bw-percent {
        type decimal64 {
            fraction-digits 5;
            range "0..100";
        }
        units "percent";
        mandatory true;
        description
            "To be used to define the guaranteed bandwidth
             as a percentage of the available service bandwidth.";
    }
    leaf end-to-end {
        type empty;
        description
            "Used if the bandwidth reservation
             must be done on the MPLS network too.";
    }
    description
        "Bandwidth constraint on the traffic class.";
}

description
    "List of class of services.";
}

description
    "Container for list of class of services.";
}

description
    "Qos profile configuration.";
}

description
    "QoS configuration.";
This grouping defines QoS parameters for a site;
key "vpn-id";
leaf vpn-id {
  type svc-id;
  description
  "Defining a service id.";
}
leaf vpn-svc-type {
  type identityref {
    base service-type;
  }
  description
  "Service type";
}
leaf customer-name {
  type string;
  description
  "Customer name";
}
leaf svc-topo {
  type identityref {
    base vpn-topology;
  }
  description
  "Defining service topology, such as
  any-to-any, hub-spoke, etc.";
}
/*uses vpn-service-cloud-access;*/

/*container global-l2cp-control {
  if-feature L2CP-control;
  leaf stp-rstp-mstp {
    type control-mode;
    description
    "STP/RSTP/MSTP protocol type applicable to all UNIs";
  }
  leaf pause {
    type control-mode;
    description
    "Pause protocol type applicable to all UNIs ";
  }
  leaf lldp {
    type boolean;
    description
    "LLDP protocol type applicable to all UNIs ";
  }
  description
  "Container of L2CP control global configurations";
}*/
uses status-params;
uses site-service;
uses vpn-service-multicast;
uses vpn-extranet;
uses svc-mtu-grouping;
uses svc-preservation-grouping;
uses svc-transport-encapsulation;
container vpn-nodes {
  list vpn-node {
    key "vpn-node-id ne-id";

    leaf vpn-node-id {
      type string;
      description "";
    } leaf description {
      type string;
      description "Textual description of a VPN node.";
    } leaf node-role {
      type identityref {
        base site-role;
      } default any-to-any-role;
      description "Role of the vpn-node in the IP VPN.";
    } leaf ne-id {
      type string;
      description "NE IP address";
    } container vpn-network-accesses {
      list vpn-network-access {
        key "network-access-id";
        leaf network-access-id {
          type string;
          description "Identifier of network access";
        } leaf description {
          type string;
          description "String to describe the element.";
        }
leaf remote-carrier-name {
    when "../site-type = 'enni'" {
        description "Site type = enni";
    }
    type string;
    description "Remote carrier name";
}
leaf Interface-mtu {
    type uint32;
    description "Interface MTU, it is also known as the maximum transmission unit or maximum frame size. When a frame is larger than the MTU, it is broken down, or fragmented, into smaller pieces by the network protocol to accommodate the MTU of the network";
}
uses status-params;
uses access-diversity;
uses ethernet-connection-grouping;
uses availability-grouping;
uses service-grouping;
uses svc-precedence;
uses B-U-M-grouping;
uses ethernet-svc-oam-grouping;
uses mac-loop-prevention-grouping;
uses acl-grouping;
uses mac-addr-limit-grouping;
uses site-bearer-params;
description "List of VPN Network Accesses.";
}
description "List of VPN Nodes."
}
description "Container of VPN Nodes."
}
description "List of vpn-svc"
}
description "Container of port configurations"
}
description "Container for L2VPN service";
}
"Container for VPN services."

} }

Figure 6

6. Acknowledgements

T

7. Contributors

Daniel King
Old Dog Consulting
Email: daniel@olddog.co.uk

8. IANA Considerations

This document requests IANA to register the following URI in the "ns" subregistry within the "IETF XML Registry" [RFC3688]:


Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

This document requests IANA to register the following YANG module in the "YANG Module Names" subregistry [RFC6020] within the "YANG Parameters" registry.

name: ietf-l2vpn-ntw


maintained by IANA: N

prefix: l2nm

reference: RFC XXXX

9. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement
secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8466].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

The ietf-l2vpn-ntw module is used to manage L2 VPNs in a service provider backbone network. Hence, the module can be used to request, modify, or retrieve L2VPN services. There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes MAY be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) and delete operations to these data nodes without proper protection or authentication can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability in the ietf-l2vpn-ntw module:

- **vpn-service**: An attacker who is able to access network nodes can undertake various attacks, such as deleting a running L2 VPN Service, interrupting all the traffic of a client.

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

- **customer-name**: An attacker can retrieve privacy-related information which can be used to track a customer. Disclosing such information may be considered as a violation of the customer-provider trust relationship.

10. References

10.1. Normative References


10.2. Informative References


Authors’ Addresses

Samier Barguil (editor)
Telefonica
Madrid
ES
Email: samier.barguilgiraldo.ext@telefonica.com

Oscar Gonzalez de Dios (editor)
Telefonica
Madrid
ES
Email: oscar.gonzalezdedios@telefonica.com

Victor Lopez
Telefonica
Madrid
ES
Email: victor.lopezalvarez@telefonica.com

Luis Angel Munoz
Vodafone
ES
Email: luis-angel.munoz@vodafone.com

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