Abstract

This document provides a YANG data model for SD-WAN VPN service. A SD-WAN VPN service is a service offered by a Service Provider network to provide an overlay connectivity between different locations of a customer network or between a customer network and an external network, such as Internet or Private Cloud network. The model can be utilized by an service orchestrator of a Service Provider to initiate a connectivity request.

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 September 11, 2019.

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.
By comparison with conventional PE-based VPN service defined in [RFC8299] and [RFC8446], the SD-WAN VPN is a type of CE-based VPN which uses the Internet or a PE based VPN as underlay connectivity service. SD-WAN uses an overlay-based approach to provide the flexibility of adding, removing, or moving services without dependence of the underlay network.

Besides being a CE-based overlay service, a SD-WAN VPN Service has the following characteristics:

- Hybrid WAN accesses: The CE could connect to variety of Internet access, including fiber, cable, DSL-based, WiFi, or 4G/Long Term Evolution (LTE) access, which implies wider reachability and
shorter provisioning cycles. It can also use private VPN connectivity defined in [RFC4364] or [RFC4664] to take advantage of better performance.

- Policy based traffic forwarding: SD-WAN VPN can provide optimizing forwarding from a network scope and deploy service as needed. Specifically, it can apply policies to prioritize traffic for diverse applications used in enterprises, such as VoIP calling, videoconferencing, streaming media etc. depending different business needs.

- Centralized service management and orchestration: The CE router is usually managed by the provider; in addition, the SP allows customers to access the CE for configuration/monitoring purposes, so a portal can enable the customer to modify the SD-WAN VPN service such as configuring application policies or adding a new site.

This draft specifies the SD-WAN VPN service YANG model which is modeled from a customer perspective and have been aligned with the objects identified in MEF SD-WAN service attributes draft document [MEF70]. The model parameters can be used as a input to automated control and configuration applications to manage SD-WAN VPN services.

1.1. Terminology

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

1.2. Definitions

CE Device: Customer Edge Device , as per Provider Provisioned VPN Terminology [RFC4026].

PE Device: Provider Edge Device, as per Provider Provisioned VPN Terminology [RFC4026]

CE-based VPN: Refers to Provider Provisioned VPN Terminology [RFC4026]

PE-Based VPNs: Refers to Provider Provisioned VPN Terminology [RFC4026]

SD-WAN: An automated, programmatic approach to managing enterprise network connectivity and circuit usage. It extends software-defined networking (SDN) into an application that businesses can use to quickly create a smart "hybrid WAN"- a WAN that comprises business-
grade IP VPN, broadband Internet, and wireless services. SD-WAN is also deemed as extended CE-based VPN.

Underlay network: The network that provides the connectivity among SD-WAN VPN sites and that the customer network packets are tunneled over. The underlay network does not need to be aware that it is carrying overlay customer network packets. Addresses on the underlay network appear as "outer addresses" in encapsulated overlay packets. In general, the underlay network can use a completely different protocol (and address family) from that of the overlay network.

Overlay network: A virtual network in which the separation of customer networks is hidden from the underlying physical infrastructure. That is, the underlying transport network does not need to know about customer separation to correctly forward traffic. IPsec tunnels [RFC6071] is an example of an L3 overlay network.

2. High Level Overview of SD-WAN VPN Service

From a customer perspective, an example of SD-WAN VPN network is shown in figure 1.
As shown in figure 1, the SD-WAN network is composed of a set of sites, which are connected through Internet or MPLS VPN.

Within each site, a CE is connected with customer’s network on one side, and is also connected to Internet or private WAN or both on the other side. The customer networks could be L2 or L3 network. For the WAN side, Internet provides ubiquitous IP connectivity via access network like Broadband access or LTE access, while MPLS WAN, like conventional VPN, provides secure and committed connectivity while attached. The demarcation point (i.e., UNI) between the customer and the SP is placed between customer nodes and the CE device.
Additionally, a site could deploy one or more CEs to improve availability.

The establishment of the SD-WAN VPN is done at each CE device, with various IP tunneling options (e.g., Generic Routing Encapsulation (GRE) [RFC2784], and IPSec [RFC6071]) could be used, and the specific definition is out of scope of this document. Either Internet or private WAN is regarded as underlay of the tunneling, the communication between Customer Network of the four sites, known as the overlay network, is agnostic of the underlying network infrastructure within the SP.

Besides connectivity between the sites, the subset of sites could also provide direct Internet connectivity, cloud network connectivity or conventional MPLS VPN connectivity.

3. Service Data Model Usage

The SD-WAN VPN service model provides an abstracted interface to request, configure, and manage the components of an SD-WAN VPN service. The model is used by a customer to request connectivity and other services from an SP.

A typical usage for this model is as an input to an orchestration layer that is responsible for service management. The Metro Ethernet Forum (MEF) [MEF55] has developed a LSO (Lifecycle Service Orchestration) Reference Architecture and Framework architecture to automate network management and operations for service provider with a SP’s SOF (Service Orchestration Functionality), which are used for orchestrating/automating the lifecycle of end-to-end service. The SD-WAN Managed service is one of the services that LSO will support.
For a SD-WAN VPN to be established under the SP’s control, the customer informs the Service Provider of which sites should become part of the requested VPN and what types of services the VPN will provide. And then the SP configures and updates the service base on the service model and the available resources derived from the SD-WAN controller, and then provisions and manages the customer’s VPN through the SD-WAN controller. How the SD-WAN controller to control and manage the CEs is out of scope of the document.

4. Design of the Data Model

The elements of the SD-WAN VPN YANG model in this document have been aligned with the objects identified in MEF SD-WAN service attributes [MEF70], but with IETF compliant terminology. The SD-WAN VPN Services are specified by three major nodes:
1. vpn: Each list node represents an end-to-end connection between two or more customer locations, which is an association of vpn-endpoints reference to site nodes.

2. sites: This list is used to indicate sites that are involved to join the SD-WAN VPN service in different geographic locations of a customer network.

3. vpn-endpoint: The endpoint list is under the vpn list, which indicates per site policy parameters pertaining to VPN are added.

4.1. SD-WAN VPN

The "sdwpn-vpn" list item contains service parameters that apply to a SD-WAN VPN, which is further specified by the following ones:

- The "vpn-id" leaf is under the vpn-service list, which refers to an internal reference for this VPN service.

- The "performance-objectives" container refers the performance-related properties of a SD-WAN VPN that can be measured. System uptime is the only one performance objective currently, which indicates the proportion of time, during a given time period, that the service is working from the customer perspective. Three parameters are defined, including the start time of the evaluation, the time interval of the evaluation, and the service uptime defined by the percentage.

- The "reserved-ipaddress" container refers to the IP Prefixes need to be agreed for Service Provider management purposes, such as diagnostics so as to ensure they are not overlapping with IP Prefixes used by the customer network.

4.1.1. VPN Endpoint

The SD-WAN VPN End Point is the logical point associated with a particular site. The two main functions of the endpoint are:

- The association of a VPN with a Site.

- Per site application based policy can be enforced.

4.1.2. Application Classification and Policy Map

The model has defined the following components to describe application-based policy.
"Application flow: an order list for IP packets from site or to the site to match. Three parameters are further specified which are application ID, application criteria list and application group.

* "application-id": is under the vpn-service list, which refers to an internal reference for this application service.

* "application-criteria": is under the "application list", which describes a set of characteristics of the packet stream that can be identified at the site, including standard layer 2 and layer 3 fields such as addresses, ports, and protocols.

* "application-group": "app-group-id" refers to an internal reference for this application group service, which describes the application categories, e.g. VOIP, email, games etc.

"policy": is a rule list. At present, path selection policy, QoS bandwidth policy and Internet local break out policy have been defined. A policy can be assigned to an application group or an application.

* path selection policies: primary-backup, billing-policy and encryption policy can be applied to the application.

* The "internet-access" container internet access option, which include local break-out for Internet access or alternative route for the traffic.

* The "qos-bandwidth" policy container is used to describe parameter to guarantee bandwidth for specific traffic flowing through a VPN connection. It has two categories parameters, including traffic rate limit and time for evaluation.

"application-group-policy-map": the list specifies the mapping of application group names and their associated policy names. The policy assignment to application group serves two purposes: first, a policy can be applied to all members of the application flow group; second, it allows application flows in the group to share bandwidth resources.

"endpoint-policy-map": the policy assignment is under "endpoints" list, which specifies the mapping of application names and their associated policy names. Each Application Flow can have an explicit policy assignment that supersedes the group policy.
4.2. Site

A site represents a customer office located at a specific location. The "sites" container specifies three main parameters:

- "site-id": uniquely identifies the site within the overall network infrastructure.
- "lan-accesses": specifies the customer network access link parameters. A "site" is composed of at least one "lan-access" and, in the case of multihoming, may have multiple links.

```
+---------------------------------+
|              site               |
|    |   |            |     |     |
|    |   |            |     |     |
|   LAN1 LAN2         LAN3 LAN4   |
|  +--------+       +--------+    |
|  |        |       |        |    |
|  |Device 1|       |Device 2|    |
|  +--------+       +--------+    |
+---------------------------------+
```

The "lan-access" consists of the following categories of parameters:

- "bearer": defines requirements of the attachment (below Layer 3), bearer type including Ethernet and etc..
- "device-type": specifies the device type, including physical or virtual device.
- IP Connection: defines Layer 3 parameters of the attachment, including IPv4 connection parameters and IPv4 connection parameters respectively.

5. Modules Tree Structure

This document defines sd-wan-vpn yang data model.

```
module: ietf-sdwan-vpn-svc
  +--rw sdwan-vpn-svc
    +--rw vpn-services
      +--rw vpn-service* [vpn-id]
      |  +--rw vpn-id svc-id
      |  +--rw performance-objective
      |      +--rw start-time? yang:date-and-time
```

```yang
++-rw duration? string
++-rw uptime-objective
    ++-rw duration? decimal64
++-rw reserved-prefixes
    ++-rw prefix* inet:ip-prefix
++-rw applications
    ++-rw application* [app-id]
        ++-rw app-id svc-id
            ++-rw ac* [name]
                +--rw name string
                ++-rw (match-type)?
                    +--:(match-flow)
                        +--rw ethertype? uint16
                        +--rw cvlan? uint8
                        +--rw ipv4-src-prefix?
                            +-- inet:ipv4-prefix
                        +--rw ipv4-dst-prefix?
                            +-- inet:ipv4-prefix
                        +--rw 14-src-port?
                            +-- inet:port-number
                        +--rw 14-dst-port?
                            +-- inet:port-number
                        +--rw ipv6-src-prefix?
                            +-- inet:ipv6-prefix
                        +--rw ipv6-dst-prefix?
                            +-- inet:ipv6-prefix
                        +--rw protocol-field? union
                            +--:(match-application)
                                +--rw match-application? identityref
                        +--rw match-flow
                            +--rw ethertype? uint16
                            +--rw cvlan? uint8
                            +--rw ipv4-src-prefix?
                                +-- inet:ipv4-prefix
                            +--rw ipv4-dst-prefix?
                                +-- inet:ipv4-prefix
                            +--rw 14-src-port?
                                +-- inet:port-number
                            +--rw 14-dst-port?
                                +-- inet:port-number
                            +--rw ipv6-src-prefix?
                                +-- inet:ipv6-prefix
                            +--rw ipv6-dst-prefix?
                                +-- inet:ipv6-prefix
                        +--rw protocol-field? union
                            +--:(match-application)
            +--:(match-application)
            +--:(match-flow)
            +--:(match-application)
++-rw application-group* [app-group-id]
    ++-rw app-group-id svc-id
        ++-rw app-id*
            -> ../../applications/application/app-id
++-rw policy* [policy-id]
    ++-rw policy-id svc-id
    ++-rw direction? enumeration
    ++-rw criterias* [pc-name]
        ++-rw pc-name string
        ++-rw (policy-type)?
            +--:(encryption)
                +--rw enable? boolean
            +--:(public-private)
                +--rw underlay-values? enumeration
            +--:(internet-breakout)
                +--rw internet-policy
                    +--rw local-breakout? boolean
                    +--rw alter-route? boolean
```
+--:(billing-method)
  |  +--rw billing-values?  enumeration
+-rw path-values
  +--rw overlay-values?  enumeration
  +--rw sla-values
    +--rw latency?  uint32
    +--rw jitter?  uint32
    +--rw packet-loss-rate?  uint32
+-rw app-group-policy-map
  +--rw mapping* [app-group-id]
    |  +--rw app-group-id
    |  -> ../../../application-group/app-group-id
    +--rw policy-id?  leafref
    +--rw policy-id?  leafref
+-rw app-group-policy-map
  +--rw endpoints* [endpoint-id]
    +--rw endpoint-id  svc-id
    +--rw site-attachment
      |  +--rw site-id?
      |  -> /sdwan-vpn-svc/sites/site/site-id
    +--rw endpoint-policy-map
      +--rw app-policy* [app-id]
        +--rw app-id  leafref
        +--rw policy-id?  leafref
+-rw sites
  +--rw site* [site-id]
    +--rw site-id  svc-id
    +--rw device-type?  device-type
  +--rw lan-access* [name]
    +--rw name  string
    +--rw l2-technology
      +--rw l2-type?  identityref
      +--rw untagged-interface
        +--rw speed?  uint32
        +--rw mode?  neg-mode
      +--rw tagged-interface
        +--rw type?  identityref
        +--rw dot1q-vlan-tagged
          +--rw tg-type?  identityref
          +--rw cvlan-id  uint16
    +--rw ip-connection
      +--rw ipv4
        +--rw address-allocation-type?  identityref
        +--rw dhcp
+rw primary-subnet
  +--rw ip-prefix?
    |  inet:ipv4-prefix
  +--rw default-router?  inet:ip-address
  +--rw provider-addresses*
    |  inet:ipv4-address
  +--rw subscriber-address?  inet:ip-address
  +--rw reserved-ip-prefix*  inet:ip-prefix
+rw secondary-subnet* [ip-prefix]
  +--rw ip-prefix
    |  inet:ipv4-prefix
  +--rw provider-addresses*
    |  inet:ipv4-address
  +--rw reserved-ip-prefix*
    |  inet:ipv4-prefix
+rw static
  +--rw primary-subnet
    |  +--rw ip-prefix?
      |     inet:ipv4-prefix
    |  +--rw default-router?  inet:ip-address
    |  +--rw provider-addresses*
      |     inet:ipv4-address
    |  +--rw subscriber-address?  inet:ip-address
    |  +--rw reserved-ip-prefix*  inet:ip-prefix
  +--rw secondary-subnet* [ip-prefix]
    +--rw ip-prefix
      |  inet:ipv4-prefix
    +--rw provider-addresses*
      |  inet:ipv4-address
    +--rw reserved-ip-prefix*
      |  inet:ipv4-prefix
+rw ipv6
  +--rw address-allocation-type?  identityref
  +--rw dhcp
    +--rw subnet* [ip-prefix]
      +--rw ip-prefix
        |  inet:ipv6-prefix
      +--rw provider-addresses*
        |  inet:ipv6-address
      +--rw reserved-ip-prefix*
        |  inet:ipv6-prefix
  +--rw slaac
    +--rw subnet* [ip-prefix]
      +--rw ip-prefix
        |  inet:ipv6-prefix
      +--rw provider-addresses*
        |  inet:ipv6-address
      +--rw reserved-ip-prefix*
6. YANG Modules

<CODE BEGINS> file "ietf-sdwan-vpn-svc@2019-03-10.yang"

module ietf-sdwan-vpn-svc {
  yang-version 1.1;
  prefix sdwan-vpn-svc;

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

  organization
    "IETF foo Working Group.";
  contact
    "WG List: foo@ietf.org
      Editor: ";
  description
    "The YANG module defines a generic service configuration
    model for SD-WAN VPN.";

  revision 2019-03-10 {
    description
      "Initial revision";
    reference "A YANG Data Model for SD-WAN VPN.";
  }

  typedef svc-id {
    type string;
    description
      "Type definition for servicer identifier";
  }

<CODE ENDS>
typedef address-family {
  type enumeration {
    enum ipv4 {
      description
      "IPv4 address family.";
    }
    enum ipv6 {
      description
      "IPv6 address family.";
    }
  }
  description
  "Defines a type for the address family.";
}

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";
}

typedef device-type {
  type enumeration {
    enum physical {
      description
      "Physical device";
    }
    enum virtual {
      description
      "Virtual device";
    }
  }
  description
  "Defines device types.";
}

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 eth-inf-type {
  description
    "Identity of the Ethernet interface type.";
}

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

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

identity lag {
  base eth-inf-type;
  description
    "Identity of the LAG interface type.";
}

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

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

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

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

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

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

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

identity 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-and-Spoke VPN topology.";
identity site-role {
    description
    "Site Role in a VPN topology ";
}

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

identity hub {
    base site-role;
    description
    "Hub Role in Hub-and-Spoke IP VPN.";
}

identity spoke {
    base site-role;
    description
    "Spoke Role in Hub-and-Spoke IP VPN.";
}

identity access-type {
    description
    "Access type of a site in a connection to a customer network or WAN network";
}

identity ge {
    base access-type;
    description
    "GE";
}

identity ef {
    base access-type;
    description
    "EF";
}

identity xge {
    base access-type;
    description
    "XGE";
}
identity lte {
    base access-type;
    description
        "LTE";
}

identity xdsl-atm {
    base access-type;
    description
        "xDSL(ATM)";
}

identity xdsl-ptm {
    base access-type;
    description
        "xDSL(PTM)";
}

identity routing-protocol-type {
    description
        "Base identity for routing protocol type.";
}

identity ospf {
    base routing-protocol-type;
    description
        "Identity for OSPF protocol type.";
}

identity bgp {
    base routing-protocol-type;
    description
        "Identity for BGP protocol type.";
}

identity static {
    base routing-protocol-type;
    description
        "Identity for static routing protocol type.";
}

identity address-allocation-type {
    description
        "Base identity for address-allocation-type for PE-CE link.";
}

identity dhcp {
    base address-allocation-type;
description
"Provider network provides DHCP service to customer."
}

identity static-address {
    base address-allocation-type;
    description
"Provider-to-customer addressing is static.";
}

identity slaac {
    base address-allocation-type;
    description
"Use IPv6 SLAAC.";
}

identity ll-only {
    base address-allocation-type;
    description
"Use IPv6 Link Local.";
}

identity traffic-direction {
    description
"Base identity for traffic direction";
}

identity inbound {
    base traffic-direction;
    description
"Identity for inbound";
}

identity outbound {
    base traffic-direction;
    description
"Identity for outbound";
}

identity both {
    base traffic-direction;
    description
"Identity for both";
}

identity traffic-action {
    description
"Base identity for traffic action";
}
identity permit {
    base traffic-action;
    description
        "Identity for permit action";
}

identity deny {
    base traffic-action;
    description
        "Identity for deny action";
}

identity bd-limit-type {
    description
        "Base identity for bd limit type";
}

identity percent {
    base bd-limit-type;
    description
        "Identity for percent";
}

identity value {
    base bd-limit-type;
    description
        "Identity for value";
}

identity protocol-type {
    description
        "Base identity for protocol field type.";
}

identity tcp {
    base protocol-type;
    description
        "TCP protocol type.";
}

identity udp {
    base protocol-type;
    description
        "UDP protocol type.";
}
identity icmp {
    base protocol-type;
    description
        "ICMP protocol type.";
}

identity icmp6 {
    base protocol-type;
    description
        "ICMPv6 protocol type.";
}

identity gre {
    base protocol-type;
    description
        "GRE protocol type.";
}

identity ipip {
    base protocol-type;
    description
        "IP-in-IP protocol type.";
}

identity hop-by-hop {
    base protocol-type;
    description
        "Hop-by-Hop IPv6 header type.";
}

identity routing {
    base protocol-type;
    description
        "Routing IPv6 header type.";
}

identity esp {
    base protocol-type;
    description
        "ESP header type.";
}

identity ah {
    base protocol-type;
    description
        "AH header type.";
}
grouping vpn-endpoint {
  leaf endpoint-id {
    type svc-id;
    description
    "Identity for the vpn endpoint";
  }
  container site-attachment {
    leaf site-id {
      type leafref {
        path "/sdwan-vpn-svc/sites/site/site-id";
      }
      description
      "Defines site id attached.";
      }
      description
      "Defines site attachment to a vpn endpoint.";
    }
  container endpoint-policy-map {
    list app-policy {
      key "app-id";
      leaf app-id {
        type leafref {
          path "/sdwan-vpn-svc/vpn-services/vpn-service/applications/application/app-id";
        }
        description
        "Identity for application";
        }
      leaf policy-id {
        type leafref {
          path "/sdwan-vpn-svc/vpn-services/vpn-service/policy/policy-id";
        }
        description
        "Identity for value";
        }
        description
        "list for application policy";
        }
        description
        "Identity for policy maps";
        }
        description
        "grouping for vpn endpoint";
    }
  }
}

grouping flow-definition {
  container match-flow {
    leaf ethertype {
      type uint16;
    }
  }
}

description
   "Ethertype value, e.g. 0800 for IPv4.";
}
leaf cvlan {
    type uint8 {
        range "0..7";
        description
        "802.1Q matching.";
    }
}
leaf ipv4-src-prefix {
    type inet:ipv4-prefix;
    description
    "Match on IPv4 src address.";
}
leaf ipv4-dst-prefix {
    type inet:ipv4-prefix;
    description
    "Match on IPv4 dst address.";
}
leaf l4-src-port {
    type inet:port-number;
    description
    "Match on Layer 4 src port.";
}
leaf l4-dst-port {
    type inet:port-number;
    description
    "Match on Layer 4 dst port.";
}
leaf ipv6-src-prefix {
    type inet:ipv6-prefix;
    description
    "Match on IPv6 src address.";
}
leaf ipv6-dst-prefix {
    type inet:ipv6-prefix;
    description
    "Match on IPv6 dst address.";
}
leaf protocol-field {
    type union {
        type uint8;
        type identityref {
            base protocol-type;
        }
    }
    description
}
"Match on IPv4 protocol or IPv6 Next Header field."
}
description
"Describes flow-matching criteria."
}
description
"groupin flow definition.";
}
grouping application-criteria {
  list ac {
    key "name";
    ordered-by user;
    leaf name {
      type string;
      description
      "A description identifying qos classification policy rule.";
    }
    choice match-type {
      default "match-flow";
      case match-flow {
        uses flow-definition;
      }
      case match-application {
        leaf match-application {
          type identityref {
            base customer-application;
          }
          description
          "Defines the application to match.";
        }
        description
        "Choice for classification.";
      }
      description
      "List of marking rules.";
    }
    description
    "This grouping defines QoS parameters for a site.";
  }
}
grouping vpn-service {
  leaf vpn-id {
    type svc-id;
    description
    "Identity for VPN.";
}
container performance-objective {
    leaf start-time {
        type yang:date-and-time;
        description
        "start-time indicats date and time.";
    }
    leaf duration {
        type string;
        description
        "Time duration.";
    }
}

container uptime-objective {
    leaf duration {
        type decimal64 {
            fraction-digits 5;
            range "0..100";
        }
        units "percent";
        description
        "To be used to define the a percentage of the available service.";
    }
    description
    "Uptime objective.";
}

container reserved-prefixes {
    leaf-list prefix {
        type inet:ip-prefix;
        description
        "ip prefix reserved for SP managment purpose.";
    }
    description
    "ip prefix list reserved for SP managment purpose.";
}

container applications {
    list application {
        key "app-id";
        leaf app-id {
            type svc-id;
            description
            "application name";
        }
        uses application-criteria;
        description
        "application";
    }

"list for application"
}

description
"container for application"

list application-group {
  key "app-group-id";
  leaf app-group-id {
    type svc-id;
    description
      "application name";
  }
  leaf-list app-id {
    type leafref {
      path "./../applications/application/app-id";
    }
    description
      "application member list in an application group";
  }
  description
    "list for application group"
}

list policy {
  key "policy-id";
  leaf policy-id {
    type svc-id;
    description
      "Policy names";
  }
  leaf direction {
    type enumeration {
      enum inbound {
        description
          "specify the wan-to-site direction to which the policy
          criteria is applied";
      }
      enum outbound {
        description
          "specify the site-to-wan direction to which the policy
          criteria is applied";
      }
      enum both {
        description
          "specify both the site-to-wan or wan-to-site direction to
          which the policy criteria is applied";
      }
    }
  }
  description
"list for policy";
}

"Traffic direction";
}
list criterias {
  key "pc-name";
  leaf pc-name {
    type string;
    description
      "Policy criteria name";
  }
  choice policy-type {
    case encryption {
      leaf enable {
        type boolean;
        description
          "yes,no.";
      }
      description
        "TVC encrypted or not.";
    }
    case public-private {
      leaf underlay-values {
        type enumeration {
          enum private-only {
            description
              "The private WAN underlay is specified.";
          }
          enum public-only {
            description
              "The public WAN underlay is specified.";
          }
          enum either {
            description
              "Both public WAN or private WAN could be used";
          }
        }
        description
          "yes,no,either.";
      }
      description
        "public-private.";
    }
    case internet-breakout {
      container internet-policy {
        leaf local-breakout {
          type boolean;
          description
            "indicates whether the Application Flow should be
             routed directly to the Internet using Local Internet
leaf alter-route {
    type boolean;
    description
        "whether an alternate route to the Internet can be used.";
}

description
    "lib,alt.";

description
    "lib,alt.";

case billing-method {
    leaf billing-values {
        type enumeration {
            enum flat-only {
                description
                    "Only flat-rate underlay could be used for the traffic.";
            }
            enum either {
                description
                    "Either flat-rate or usage based underlay could be used for the traffic.";
            }
        }
        description
            "billing policy.";
    }
}
case primary-backup {
    container path-values {
        leaf overlay-values {
            type enumeration {
                enum primary {
                    description
                        "Only the primary tunnel overlay could be used for the traffic.";
                }
                enum either {
                    description
                        "Either the primary or backup overlay tunnel could be used for the traffic.";
                }
            }
            description
                "Breakout. It can have values Yes and No.";
        }
    }
}
"overlay connection as Primary or both Primary and Backup."
}
container sla-values {
    leaf latency {
        type uint32;
        description
            "latency";
    }
    leaf jitter {
        type uint32;
        description
            "jitter";
    }
    leaf packet-loss-rate {
        type uint32;
        description
            "packet loss rate";
    }
    description
        "traffic sla";
    }
    description
        "path values";
    }
    description
        "primary-backup policy";
}
case bandwidth {
    container bandwidth-values {
        leaf commit {
            type uint32;
            description
                "CIR";
        }
        leaf max {
            type uint32;
            description
                "max speed ";
        }
        leaf time {
            type uint32;
            description
                "the averaging period (in milliseconds) for
determining the information rates ";
        }
    }
    description
        "Container for value";
}  

description  
"case for bandwidth policy.";
} 

description  
"Choice for policy criteria.";
} 

description  
"List for pc criteria";
} 

description  
"List for policy";
} 

container app-group-policy-map { 
  list mapping { 
    key "app-group-id"; 
    leaf app-group-id { 
      type leafref { 
        path "/"../../../application-group/app-group-id"; 
      } 
      description  
"List for policy";
    } 

    leaf policy-id { 
      type leafref { 
        path "/"../../../policy/policy-id"; 
      } 
      description  
"policy reference";
    } 

    description  
"List for policy mapping";
    } 

    description  
"container for policy mapping ";
  list endpoints { 
    key "endpoint-id"; 
    uses vpn-endpoint; 
    description  
"List of endpoints.";
    } 

    description  
"List of vpn service";
  }

grouping site-l2-technology { 
  container l2-technology {

  }

  
}

---

leaf l2-type {
  type identityref {
    base eth-inf-type;
  }
  default "untagged";
  description
  "Defines physical properties of an interface. By default, the
  Ethernet interface type is set to ‘untagged’.";
}

container untagged-interface {
  leaf speed {
    type uint32;
    units "mbps";
    default "10";
    description
    "Port speed.";
  }
  leaf mode {
    type neg-mode;
    default "auto-neg";
    description
    "Negotiation mode.";
  }
  description
  "Container of Untagged Interface Attributes
  configurations.";
}

container tagged-interface {
  leaf type {
    type identityref {
      base tagged-inf-type;
    }
    default "dot1q";
    description
    "Tagged interface type. By default, the Tagged interface type is dot1q interface. ";
  }
  container dot1q-vlan-tagged {
    leaf tg-type {
      type identityref {
        base tag-type;
      }
      default "c-vlan";
      description
      "TAG type. By default, Tag type is Customer-VLAN tag.";
    }
    leaf cvlan-id {
      type uint16;
    }
  }
  leaf dot1q-vlan-tagged {
    type identityref {
      base dot1q-inf-type;
    }
    default "dot1q";
    description
    "Tagged interface type. By default, the Tagged interface type is dot1q interface. ";
  }
  description
  "Container of Tagged Interface Attributes
  configurations.";
}
mandatory true;
description
"VLAN identifier.";
}
description
"Tagged interface.";
}
description
"Container for tagged Interface.";
}
description
"Container for l2 technology.";
}
description
"grouping for l2 technology.";
}

grouping site-ip-connection {
  container ip-connection {
    container ipv4 {
      leaf address-allocation-type {
        type identityref {
          base address-allocation-type;
        }
        description
        "Defines how addresses are allocated.
        If there is no value for address allocation type, then the ipv4 is not enabled.";
      }
      container dhcp {
        container primary-subnet {
          leaf ip-prefix {
            type inet:ipv4-prefix;
            description
            "IPv4 address prefix and mask length between 0 and 31, in bits.";
          }
          leaf default-router {
            type inet:ip-address;
            description
            "Address of default router.";
          }
          leaf-list provider-addresses {
            type inet:ipv4-address;
            description
            "the Service Provider IPv4 Addresses MUST be within the specified IPv4 Prefix.";
          }
        }
      }
    }
  }
}

Sun, et al.  Expires September 11, 2019    [Page 34]
leaf subscriber-address {
  type inet:ip-address;
  description
    "subscriber IPv4 Addresses: Non-empty list
     of IPv4 addresses";
}
leaf-list reserved-ip-prefix {
  type inet:ip-prefix;
  description
    "List of IPv4 Prefixes, possibly empty";
  description
    "Primary Subnet List";
}
list secondary-subnet {
  key "ip-prefix";
  leaf ip-prefix {
    type inet:ipv4-prefix;
    description
      "IPv4 address prefix and mask length between 0 and 31,
       in bits";
  }
  leaf-list provider-addresses {
    type inet:ipv4-address;
    description
      "Service Provider IPv4 Addresses: Non-empty list
       of IPv4 addresses";
  }
  leaf-list reserved-ip-prefix {
    type inet:ipv4-prefix;
    description
      "List of IPv4 Prefixes, possibly empty";
  }
  description
    "Secondary Subnet List";
  description
    "DHCP allocated addresses related parameters.";
}
container static {
  container primary-subnet {
    leaf ip-prefix {
      type inet:ipv4-prefix;
      description
        "IPv4 address prefix and mask length between 0 and 31,
         in bits.";
    }
    leaf default-router {

type inet:ip-address;
description
  "Address of default router."
}
leaf-list provider-addresses {
type inet:ipv4-address;
description
  "the Service Provider IPv4 Addresses MUST be within the specified IPv4 Prefix."
}
leaf subscriber-address {
type inet:ip-address;
description
  "subscriber IPv4 Addresses: Non-empty list of IPv4 addresses"
}
leaf-list reserved-ip-prefix {
type inet:ipv4-prefix;
description
  "List of IPv4 Prefixes, possibly empty"
}
description
  "Primary Subnet List"
}
list secondary-subnet {
  key "ip-prefix";
  leaf ip-prefix {
    type inet:ipv4-prefix;
description
      "IPv4 address prefix and mask length between 0 and 31, in bits"
  }
  leaf-list provider-addresses {
    type inet:ipv4-address;
description
      "Service Provider IPv4 Addresses: Non-empty list of IPv4 addresses"
  }
  leaf-list reserved-ip-prefix {
    type inet:ipv4-prefix;
description
      "List of IPv4 Prefixes, possibly empty"
  }
description
  "Secondary Subnet List"
}
description
  "Static configuration related parameters.";
description "IPv4-specific parameters.";
}
container ipv6 {
  leaf address-allocation-type {
    type identityref {
      base address-allocation-type;
    }
  }
  description "Defines how addresses are allocated. If there is no value for address allocation type, then the ipv6 is not enabled.";
}
container dhcp {
  list subnet {
    key "ip-prefix";
    leaf ip-prefix {
      type inet:ipv6-prefix;
      description "IPv6 address prefix and prefix length between 0 and 128";
    }
    leaf-list provider-addresses {
      type inet:ipv6-address;
      description "Non-empty list of IPv6 addresses";
    }
    leaf-list reserved-ip-prefix {
      type inet:ipv6-prefix;
      description "List of IPv6 Prefixes, possibly empty";
    }
    description "Subnet List";
  }
  description "DHCP allocated addresses related parameters.";
}
container slaac {
  list subnet {
    key "ip-prefix";
    leaf ip-prefix {
      type inet:ipv6-prefix;
      description "IPv6 address prefix and prefix length of 64 ";
    }
    leaf-list provider-addresses {
type inet:ipv6-address;

description
   "Non-empty list of IPv6 addresses";
}
leaf-list reserved-ip-prefix {
    type inet:ipv6-prefix;
    description
       "List of IPv6 Prefixes, possibly empty";
}
description
   "Subnet List";
}
description
   "DHCP allocated addresses related parameters.";
}
container static {
    list subnet {
        key "ip-prefix";
        leaf ip-prefix {
            type inet:ipv6-prefix;
            description
               "IPv6 address prefix and prefix length between 0 and 128";
        }
        leaf-list provider-addresses {
            type inet:ipv6-address;
            description
               "Non-empty list of IPv6 addresses";
        }
        leaf-list reserved-ip-prefix {
            type inet:ipv6-prefix;
            description
               "List of IPv6 Prefixes, possibly empty";
        }
        description
           "Subnet List";
    }
    leaf subscriber-address {
        type inet:ipv6-address;
        description
           "IPv6 address or Not Specified.";
    }
    description
       "Static configuration related parameters.";
    description
       "Describes IPv6 addresses used.";
}
description
  "IPv6-specific parameters."
}
}

container sdwan-vpn-svc {
  container vpn-services {
    list vpn-service {
      key "vpn-id";
      uses vpn-service;
      description
        "List for SD-WAN";
    }
    description
      "Container for SD-WAN VPN service";
  }
  container sites {
    list site {
      key "site-id";
      leaf site-id {
        type svc-id;
        description
          "Site Name";
      }
      leaf device-type {
        type device-type;
        description
          "device type";
      }
      list lan-access {
        key "name";
        leaf name {
          type string;
          description
            "lan access link name";
        }
        uses site-l2-technology;
        uses site-ip-connection;
        description
          "container for lan access";
      }
      description
        "List for site";
  }
  description
    "Container for sites";
7. Security Considerations

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

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability.

8. IANA Considerations

IANA has assigned a new URI from the "IETF XML Registry" [RFC3688].

    Registrant Contact: The IESG
    XML: N/A; the requested URI is an XML namespace.

IANA has recorded a YANG module name in the "YANG Module Names" registry [RFC6020] as follows:

    Name: ietf-sdwan-vpn-svc
    Prefix: sdwan-svc
    Reference: RFC xxxx
9. Appendix 1: MEF SD-WAN Service Attributes Terminology Mapping

The below table shows the terminology mapping. Besides the difference, the MEF defines the service attribute of the UNI or SWVC object in a parallel approach. However, in order to reflect the relevance of the parameters, the YANG model retains the parameter name but adjusts some of the structure. Additionally, in order to preserve the space for future augmentation, the model defines "lan-access" as a list, which can also accommodate the case where the current MEF service attribute restricts only one LAN access.

+-----------------------------+----------------------------------+
| IETF SD-WAN Service model   | MEF70 SD-WAN Services Term       |
+-----------------------------+----------------------------------+
| SD-WAN VPN                  | SD-WAN Virtual Connection (SWVC) |
+-----------------------------+----------------------------------+
| SD-WAN VPN Endpoint         | SWVC End Point                   |
+-----------------------------+----------------------------------+
| Site                        | User Network Interface (UNI)     |
+-----------------------------+----------------------------------+
| lan access                  | UNI Service Attributes           |
+-----------------------------+----------------------------------+

10. Appendix 2: Site Augmentation and Policy Augmentation

In some cases, a customer needs to have a whole view of site network connection which not only includes customer network but also includes WAN connectivity.

10.1. Site Augmentation

A Site node could be augmented with WAN access list to show the underlay network information.
10.2. Path Selection Policy Augmentation

For the traffic specified by the flow classification rule, traffic SLA profile related status will be collected and based on the measurement result calculated from the collected information, primary path or secondary path will be selected.

```plaintext
+--:(primary-backup)
    +--rw path-values
        +--rw overlay-values?   enumeration
        +--rw sla-values
            +--rw latency?         uint32
            +--rw jitter?          uint32
            +--rw packet-loss-rate? uint32
```

11. Acknowledgments

This work has benefited from the discussions of xxxx.

12. Contributors

The authors would like to thank Zitao Wang and Qin Wu for their major contributions to the initial modeling.
13. References

13.1. Normative References


13.2. Informative References


MEF, Ed., "SD-WAN Service Attributes and Service Description".


Authors’ Addresses

Qiong Sun
China Telecom
Beijing
China

Email: sunqiong.bri@chinatelecom.cn

Honglei Xu
China Telecom
Beijing
China

Email: sunqiong.bri@chinatelecom.cn