Operations and Management Area Working Group   Q. Sun
Internet-Draft   H. Xu
Intended status: Standards Track   China Telecom
Expires: January 4, 2020   B. Wu, Ed.
                                                 Q. Wu, Ed.
                                                     Huawei
                                              C. Eckel, Ed.
                                              Cisco Systems
                                               July 3, 2019

A YANG Data Model for SD-WAN Service Delivery
draft-sun-opsawg-sdwan-service-model-04

Abstract

   This document provides a YANG data model for an SD-WAN service. An
   SD-WAN service is a connectivity service offered by a service
   provider network to provide connectivity across different locations
   of a customer network or between a customer network and an external
   network, such as the Internet or a private/public cloud network.
   This connectivity is provided as an overlay constructed using one of
   more underlay networks. The model can be used by a service
   orchestrator of a service provider to request, configure, and manage
   the components of an SD-WAN service.

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 January 4, 2020.

Copyright Notice

Table of Contents

1. Introduction

An SD-WAN service is a connectivity service offered by a service
provider network to provide connectivity across different locations
of a customer network or between a customer network and an external
network. Compared to a conventional PE-based connectivity service as
defined in Layer 3 VPN Service Model [RFC8299] and Layer 2 VPN
Service Model [RFC8466], an SD-WAN service is a CE-based connectivity
service that uses the Internet or PE-based connectivity services as
underlay connectivity services. More specially, an SD-WAN service is
an overlay connectivity service that provides the flexibility of

adding, removing, or moving services without needing to change the
underlay networks.

Besides being an overlay service, an SD-WAN Service has the following
characteristics:

o Hybrid WAN access: The CE could connect to a variety of Internet
   access technologies, including fiber, cable, DSL-based, WiFi, or
   4G/Long Term Evolution (LTE), which implies wider reachability and
   shorter provisioning cycles. It can also use private VPN
   connectivity services defined in [RFC4364] and [RFC4664], or
   Operator Ethernet Services, as defined in [MEF51.1], to take
   advantage of better performance.

o Application based traffic forwarding: There are diverse
   applications used in enterprises, such as VoIP calling, video
   conferencing, streaming media, etc. Application traffic across
   the WAN will be forwarded based on business priorities, SLA
   requirements, or other enterprise requirements.

o Centralized service management: Subscribers of the service need to
   be provided a single point (such as a web portal) from which to
   dynamically add or modify services, such as configuring
   application policies, adding new sites, or adding new underlay
   connectivity services.

This draft specifies the SD-WAN service YANG model which is modelled
from a customer perspective. The model parameters can be used as an
input to automated control and configuration applications to manage
SD-WAN 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] .

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

PE Device: Provider Edge Device, as per 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 hybrid WAN, which comprises business-grade IP VPN,
   broadband Internet, and wireless services or multiple WANs of the
   same or different types. SD-WAN is also deemed as extended CE-based
   VPN.

   SD-WAN Controller: Refers to the abstract entity that combines
   Control Plane (CP) and Management Plane (MP) defined in SDN: Layers
   and Architecture Terminology [RFC7426], to configure, manage and
   control the CEs and other corresponding SD-WAN components.

   Underlay network: A network that provides connectivity across SD-WAN
   sites and over which customer network packets are tunnelled. An
   underlay network does not need to be aware that it is carrying
   overlay customer network packets. Addresses on an underlay network
   appear as "outer addresses" in encapsulated overlay packets. In
   general, an 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 networks do not
   need to know about customer separation to correctly forward traffic.
   IPsec tunnels [RFC6071] are an example of an L3 overlay network.

2. High Level Overview of SD-WAN Service

   From a customer perspective, an example of SD-WAN service network is
   shown in figure 1.

```
                          +-------------+
      +------------+      |    +---+    |
      | Controller +----+ |    |CN |    |    Legend:Customer Network
      +------------+    | |    +---+    |
                        | |      | site3|
                        | |    +--+--+  |
                     +--|---|CE 4 |    |
                        | |    +--+--+  |
                        |   +-------------+
                        |        |
                        +------------------ ----+
                        |     -----           |
          +--------------+   /  MPLS   \   +-----------------+
          |              | | |   WAN    |__|    |            |
          |              | |  /\         /\   \ +--+--+      |
          |              | | /    +-----+   \ |\|CE 1 +-+    |
          | +---+  +----++|/              \|/+--+--+ |  +---+|
          | |CN +--+ CE 3||               \         +--+CN ||
          | +---+  +-----+|     ------    /|\+--+--+ |  +---+|
          |              | |\  /Internet\ / |/|CE 2 +-+    |
          |              | | --|   WAN    |__/ +--+--+      |
          |        site 2| |   \         /   |   site 1    |
          +--------------+      ------     +-----------------+
                        |         |
                        |   +-------------+
                        |   |   +----+    |
                     +----|---+ CE5|    |
                        |   +----+    |
                        |site 4|      |
                        |      |      |
                        |   +---+      |
                        |   |CN |      |
                        |   +---+      |
                        +------------+
```
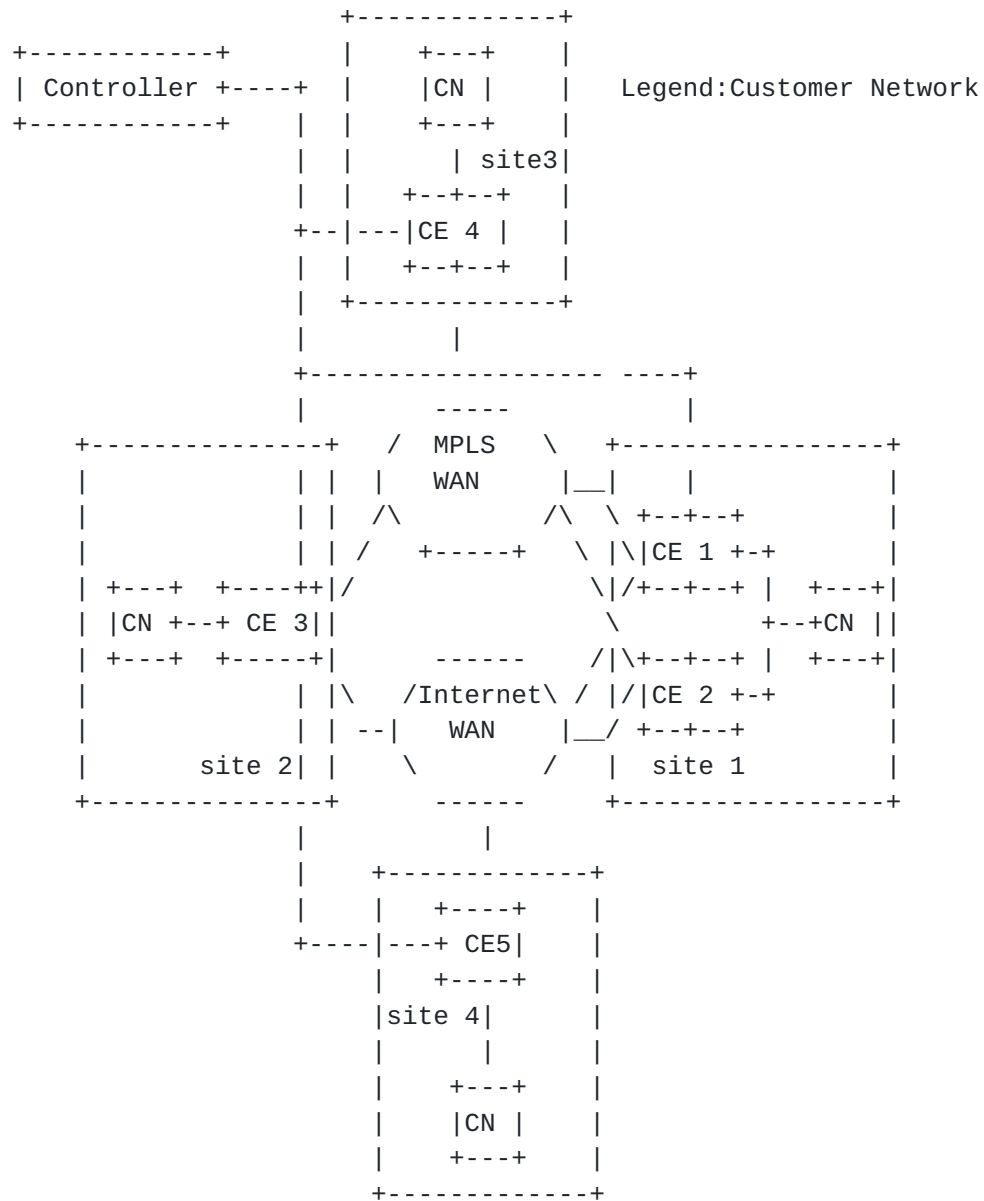
                    figure 1 SD-WAN network example

   As shown in figure 1, the SD-WAN network consists of a number 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 to private WAN, or to
   both on the other side. The customer network could be an 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. The boundary between the customer and the service
   provider is between customer node and the CE device.

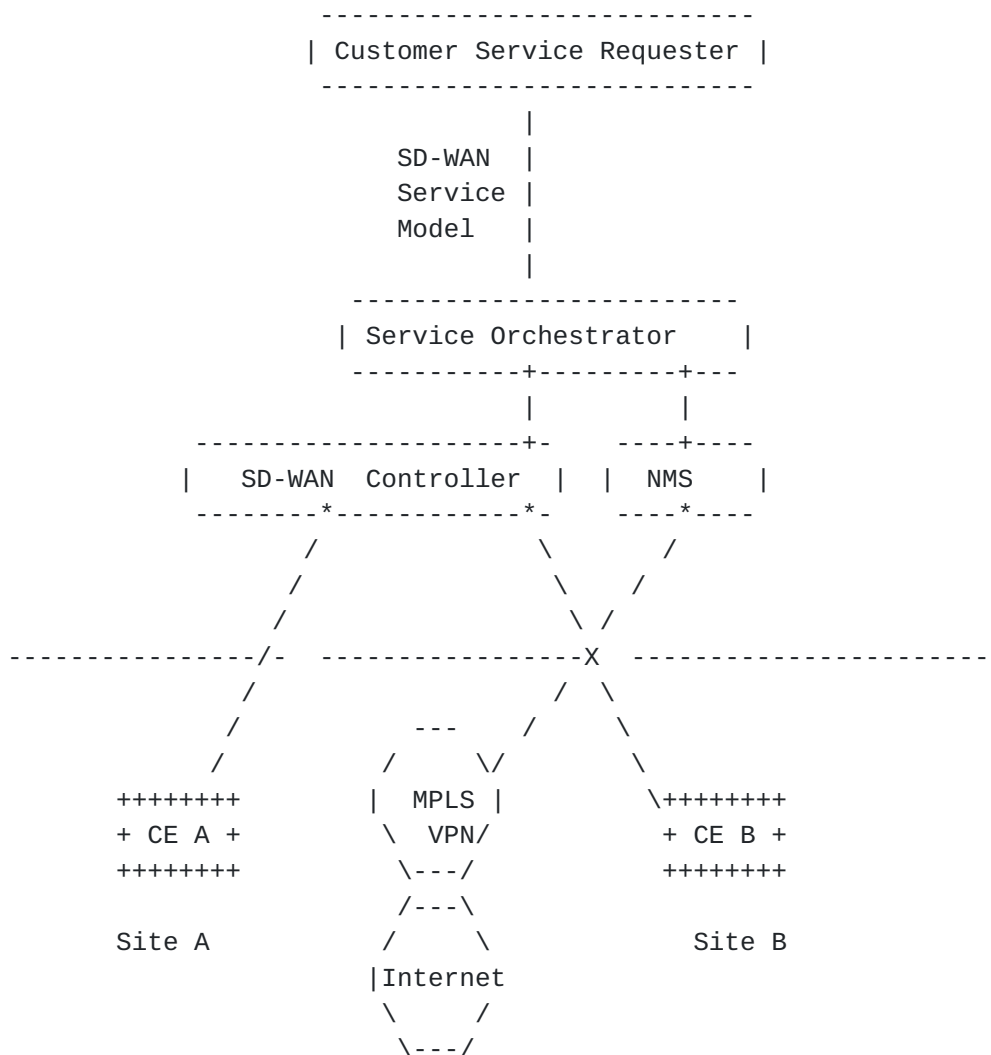Additionally, a site could deploy one or more CEs to improve availability.

The controller is a centralized entity that manages all the CEs involved in the SD-WAN. The controller could provide bootstrapping of the CEs, ongoing CE configuration, and establishment of secured tunnels between CEs to support the SD-WAN service and application policy enforcement. Various IP tunnelling options (e.g., GRE [RFC2784] and IPSec [RFC6071]), could be used depending on whether traffic from the site is across underlying private VPN or public Internet, and the specific definition is out of scope of this document.

Besides basic connectivity between the sites, the SD-WAN service could be extended by providing direct Internet connectivity, cloud network connectivity, or conventional MPLS VPN interoperability.

3. Service Data Model Usage

The SD-WAN service model provides an abstracted interface to request, configure, and manage the components of an SD-WAN service.

A typical usage for this model is as an input to a service orchestrator that is responsible for service management. Based on the user's service request, the service orchestrator can instruct the SD-WAN controller to add a new site,VPN or application policy in real-time. The orchestrator could orchestrator the other network, such as legacy MPLS VPN network to interconnect with SD-WAN network where Layer 2 VPN Service Mode [RFC8466] or Layer 3 VPN Service Model [RFC8299] could be used.

```
                    -----------------------------
                    | Customer Service Requester |
                    -----------------------------
                                 |
                         SD-WAN  |
                         Service |
                         Model   |
                                 |
                      ---------------------------
                      | Service Orchestrator    |
                       -----------+---------+---
                                  |         |
               ---------------------+-     ----+----
               |   SD-WAN  Controller |   |  NMS    |
               --------*------------*-     ----*----
                      /              \        /
                     /                \      /
                    /                  \    /
   ----------------/-  ----------------X  -----------------------
                  /                  / \
                 /          ---      /   \
                /          /   \/   /     \
          ++++++++       |  MPLS |      \++++++++
          + CE A +        \  VPN/        + CE B +
          ++++++++         \---/         ++++++++
                           /---\
          Site A          /     \            Site B
                         |Internet
                          \     /
                           \---/
```

       Reference Architecture for the Use of SD-WAN Service Model Usage

     For an SD-WAN to be established under the SP's control, the customer
     informs the Service Provider of which sites should become part of the
     requested service and what types of policy 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 service 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

     An SD-WAN service consist of two service components:

     1. SD-WAN connectivity service

   2. SD-WAN application policy service

4.1. SD-WAN connectivity service

   SD-WAN connectivity service is the basic component of the SD-WAN
   service that represents a virtual connection between two or more
   customer sites. In this model, each virtual connection is defined as
   a VPN. Each customer can have one or more VPNs, and each VPN can be
   established between a subset of sites. The association of sites and
   VPNs is modelled by VPN endpoints.

4.1.1. VPNs

   The "sdwan-vpn" list item contains service parameters that apply to
   an SD-WAN VPN. These parameters are specified as follows:

   o The "vpn-id" leaf is under the vpn-service list, and providers a
      unique ID for a VPN.

   o The "endpoints" list is under the vpn-service list. Each
      "endpoint" is a logical point associated with a site. The two
      main functions of the endpoint are the association of a VPN with a
      site and per site application based policy enforcement.

   o The "topology" leaf is under the vpn-service list, which refers to
      a specific topology of the VPN service. Different VPN connection
      topology can be used. For a VPN with a few sites, simple
      topologies such as hub-and-spoke or full-mesh can be used. For a
      large VPN, a hierarchical topology may be taken.

   o The "performance-objectives" container specifies the performance-
      related properties of an SD-WAN VPN that can be measured. System
      uptime is the only performance objective defined currently. It
      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 a percentage.

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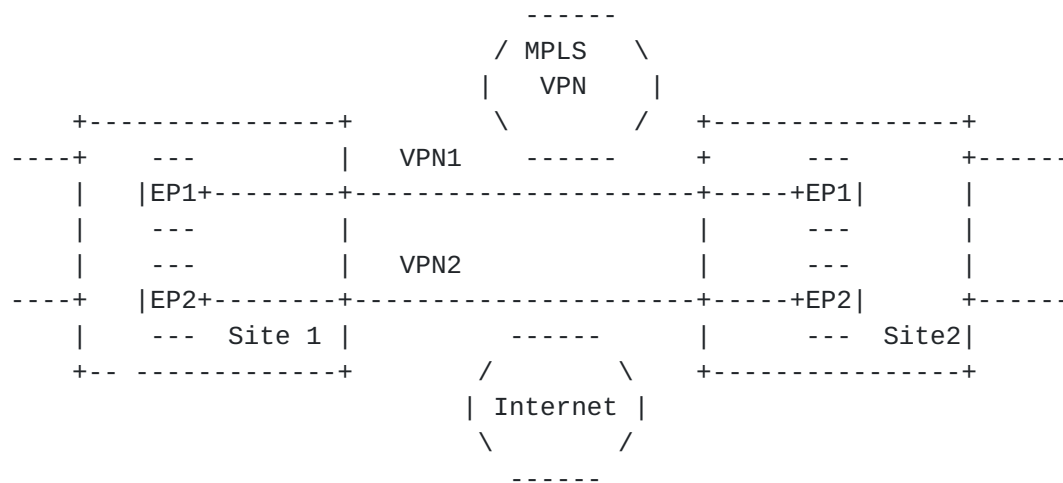   Prefixes used by the customer network.

```
                              ------
                             / MPLS   \
                             |   VPN     |
       +----------------+      \         /    +----------------+
     ----+     ---          |  VPN1    ------     +       ---       +------
       |    |EP1+--------+----------------------+-----+EP1|       |
       |     ---          |                          |      ---      |
       |     ---          |  VPN2                    |      ---      |
     ----+    |EP2+--------+----------------------+-----+EP2|      +------
       |     ---   Site 1 |        ------           |      ---   Site2|
       +-- -------------+       /         \    +----------------+
                             | Internet |
                              \         /
                               ------
```

                    figure 3 SD-WAN VPN example

4.1.2. Sites

   A site represents a customer office located at a specific geographic
   location. The "sites" container specifies the following parameters:

   o "site-id: uniquely identifies the site within the overall network
      infrastructure.

   o "device" specifies the device type (physical or virtual device)
      and the number of the devices.

   o "lan-accesses": Specifies the customer network access link
      parameters. A "site" is composed of at least one "lan-access"
      where one or more subnets can reside.The "lan-access" consists of
      the following categories of parameters:

      * "bearer": defines requirements of the attachment (below Layer
         3), bearer type including Ethernet, etc.

      * IP Connection: defines Layer 3 parameters of the attachment,
         including IPv4 connection parameters and IPv6 connection
         parameters.

   o "wan-accesses": Specifies the WAN access link parameters. A
      "site" is composed of at least one "wan-access". The WAN access
      can be further specified by access type, service provider name,
      and bandwidth of the WAN connectivity. The "wan-access" consists
      of the following categories of parameters:

      * "access-type":specifies whether the access is Broadband
         Internet, Wireless Internet or private circuit.

    *   "access-provider": specifies the service provider name.

    *   bandwidth: specifies the WAN link bandwidth including input and
        output bandwidth.

    *   "bearer": defines requirements of the attachment (below Layer
        3), bearer type including Ethernet, etc.

    *   IP Connection: defines Layer 3 parameters of the attachment,
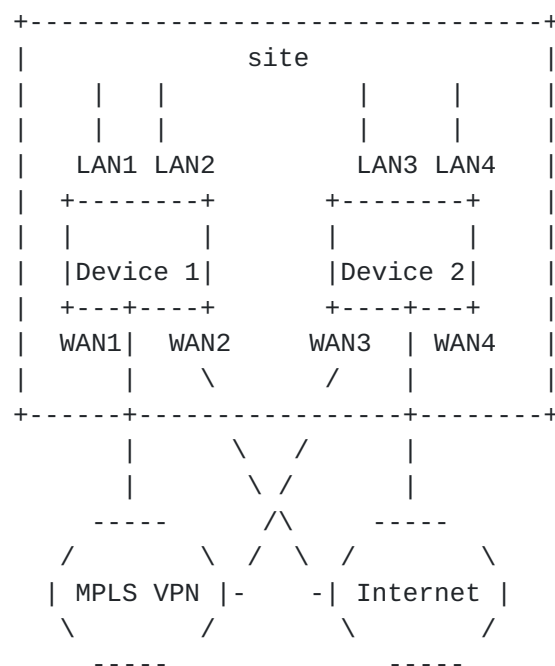   including IPv4 connection parameters and IPv6 connection
        parameters.

```
              +----------------------------------+
              |                site              |
              |    |    |              |    |     |
              |    |    |              |    |     |
              |   LAN1 LAN2         LAN3 LAN4     |
              |   +--------+       +--------+     |
              |   |        |       |        |     |
              |   |Device 1|       |Device 2|     |
              |   +---+----+       +----+---+     |
              |   WAN1|  WAN2      WAN3 | WAN4     |
              |    |    |   \       /   |          |
              +------+----------------+--------+
                    |       \   /       |
                    |        \ /        |
                 -----        /\       -----
                /        \   / \   /        \
               | MPLS VPN |-    -| Internet |
                \        /        \        /
                 -----              -----
```

                    figure 4 Site example

4.2. Application based Policy Service

   The connectivity service establishes a virtual connection for the
   enterprise network, and the Application based Policy Service is
   designed to ensure business-critical and real-time application
   experience while also ensuring the security and corporate policies.

   Typically, application policies common to each VPN can be defined and
   then enforced when traffic from a customer's network at a particular
   site is sent over the WAN.

   The application policy assignment is defined under the VPN endpoint
   container to specify the mapping of application flow name or
   application group name and their associated policy list names. If an

application flow and the application flow group in which the
Application Flow is a member are both assigned a policy at an VPN End
Point, the policy assigned to the application flow will supersedes
the group policy.

The application policy per VPN consist of three lists under the VPN
container:

o application flow list: Describes the characteristics of an
   enterprise application and is used to identify applications, e.g.,
   based on layer 3 source and destination addresses, layer 4 ports,
   layer 4 protocol, etc.

o application group list: Describes application flow aggregation,
   which is used to deliver aggregation policies, such as bandwidth
   restrictions for a group of applications.

o policy list: Defines the application's policy set. Since SD-WAN
   has more than one WAN connectivity and various encrypted or
   unencrypted overlay tunnels, there could be multiple tunnel or
   link selection combination. In this model, different path
   selection policies are combined to meet different needs based on
   application SLA, security, cost, and so on. For example, when
   different applications in a branch need to pass over the WAN,
   according to the application-aware policy requirements and the IP
   forwarding table, the Internet application or the SaaS application
   can be accessed through the Internet, and the data center FTP
   application can use the Internet encrypted tunnel as the primary
   path, and the tunnel could only be over broadband Internet instead
   of wireless internet. This policy combination is not an
   exhaustive list and could be augmented according to business
   needs.

An example of a classification of application flows is as follows:

The HTTP traffic from the 192.0.2.0/24 LAN destined for port 80
will be classified in app-id 1.

The FTP traffic from the 192.0.2.0/24 LAN destined for 203.0.113.1/32
will be classified in app-id 2.

An example of a policy list is as follows:

```
      "policy": [
        {
          "policy-id": "pol-a",
          "policy-package":
              {
                  "encryption": "false",
                  "internet-breakout": "true"
                  "public-private": "public",
                  "billing-method": "flat-only"
                  "backup": "false",
                  "bandwidth": "20","50"
              }
         },
        {
          "policy-id": "pol-b",
          "policy-package":
            {
                "encryption": "true",
                "internet-breakout": "false"
                 "public-private": "public",
                 "billing-method": "flat-only"
                 "backup": "false",
                 "bandwidth": "50","none"
                 }
        }
      ]
```

An example of an application policy list is as follows:

```
      "app-policy": [
        {
          "app-id": "1"
          "policy-id": "pol-a",
         },
        {
           "app-id": "1"
           "policy-id": "pol-b",
        }
      ]
```

5. Modules Tree Structure

   This document defines an SD-WAN service YANG data model.

```
   module: ietf-sdwan-svc
     +--rw sdwan-svc
        +--rw vpn-services
        |  +--rw vpn-service* [vpn-id]
```

```
        |        +--rw vpn-id                    svc-id
        |        +--rw topology?                 identityref
        |        +--rw performance-objective
        |        |  +--rw start-time?         yang:date-and-time
        |        |  +--rw duration?           string
        |        |  +--rw uptime-objective
        |        |     +--rw duration?   decimal64
        |        +--rw reserved-prefixes
        |        |  +--rw prefix*   inet:ip-prefix
        |        +--rw application* [app-id]
        |        |  +--rw app-id    svc-id
        |        |  +--rw ac* [name]
        |        |     +--rw name                       string
        |        |     +--rw (match-type)?
        |        |        +--:(match-flow)
        |        |        |  +--rw match-flow
        |        |        |     +--rw ethertype?        uint16
        |        |        |     +--rw cvlan?            uint8
        |        |        |     +--rw ipv4-src-prefix?  inet:ipv4-prefix
        |        |        |     +--rw ipv4-dst-prefix?  inet:ipv4-prefix
        |        |        |     +--rw l4-src-port?      inet:port-number
        |        |        |     +--rw l4-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 application-group* [app-group-id]
        |        |  +--rw app-group-id    svc-id
        |        |  +--rw app-id*         -> ../../application/app-id
        |        +--rw policy* [policy-id]
        |        |  +--rw policy-id        svc-id
        |        |  +--rw policy-package
        |        |     +--rw encryption?       enumeration
        |        |     +--rw public-private?   enumeration
        |        |     +--rw local-breakout?   boolean
        |        |     +--rw billing-method?   enumeration
        |        |     +--rw backup-path?      enumeration
        |        |     +--rw bandwidth
        |        |        +--rw commit?   uint32
        |        |        +--rw max?      uint32
        |        +--rw endpoints* [endpoint-id]
        |           +--rw endpoint-id           svc-id
        |           +--rw site-role?            identityref
        |           +--rw site-attachment
        |           |  +--rw site-id?   -> /sdwan-svc/sites/site/site-id
        |           +--rw endpoint-policy-map
        |              +--rw app-group-policy* [app-group-id]
```

```
|            |  +--rw app-group-id    leafref
|            |  +--rw policy-id?       leafref
|          +--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* [name]
      |  +--rw name    string
      |  +--rw type?   identityref
      +--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 priority-tagged
      |  |  |     +--rw tag-type?   identityref
      |  |  +--rw l2-mtu?                 uint32
      |  +--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*
      |         |                inet:ipv6-prefix
      |         +--rw static
      |            +--rw subnet* [ip-prefix]
      |            |  +--rw ip-prefix
      |            |  |         inet:ipv6-prefix
      |            |  +--rw provider-addresses*
      |            |  |         inet:ipv6-address
      |            |  +--rw reserved-ip-prefix*
      |            |            inet:ipv6-prefix
      |            +--rw subscriber-address?   inet:ipv6-address
      +--rw wan-access* [name]
         +--rw name                string
         +--rw access-type?        identityref
         +--rw access-provider?    string
         +--rw bandwidth
         | +--rw input-bandwidth?    uint64
         | +--rw output-bandwidth?   uint64
         +--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 priority-tagged
                  | |    +--rw tag-type?   identityref
                  | +--rw l2-mtu?                 uint32
                  +--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*
                          |               inet:ipv6-prefix
                          +--rw static
                             +--rw subnet* [ip-prefix]
                             |   +--rw ip-prefix
                             |   |        inet:ipv6-prefix
                             |   +--rw provider-addresses*
                             |   |        inet:ipv6-address
                             |   +--rw reserved-ip-prefix*
                             |            inet:ipv6-prefix
                             +--rw subscriber-address?    inet:ipv6-address
```

6. YANG Modules

```
<CODE BEGINS> file "ietf-sdwan-svc@2019-06-06.yang"

module ietf-sdwan-svc {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-sdwan-svc";
  prefix sdwan-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 Managed SD-WAN.";

  revision 2019-06-06 {
    description
      "Initial revision";
    reference "A YANG Data Model for SD-WAN service.";
  }

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

  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 device-type {
    description
      "Base identity for device type.";
  }

  identity virtual-ce {
    base device-type;
    description
      "Identity for virtual-ce.";
  }

  identity physical-ce {
    base device-type;
    description
      "Identity for physical-ce.";
  }

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

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

```
   identity priority-tagged {
     base tagged-inf-type;
     description
       "This identity the priority-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 different WAN";
  }

  identity commodity {
    base access-type;
    description
      "Internet access";
  }

  identity cellular {
    base access-type;
    description
      "Refers to a subset of 3G/4G/LTE and 5G";
  }

  identity private {
    base access-type;
    description
      "Refers to private circuits such as Ethernet, T1, etc";
  }

  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";
    }
    leaf site-role {
      type identityref {
        base site-role;
      }
      default "any-to-any-role";
      description
        "Role of the site in the VPN.";
    }
    container site-attachment {
      leaf site-id {
        type leafref {
          path "/sdwan-svc/sites/site/site-id";
        }
        description
          "Defines site id attached.";
      }
      description
        "Defines site attachment to a vpn endpoint.";
    }
    container endpoint-policy-map {
      list app-group-policy {
        key "app-group-id";
        leaf app-group-id {
          type leafref {
            path "/sdwan-svc/vpn-services/vpn-service"+
            "/application-group/app-group-id";
          }
          description
            "Identity for application";
        }
        leaf policy-id {
          type leafref {
            path "/sdwan-svc/vpn-services/vpn-service/policy/policy-id";
          }
          description
            "Identity for value";
        }
        description
          "list for application group policy";
      }
```

```
      list app-policy {
        key "app-id";
        leaf app-id {
          type leafref {
            path "/sdwan-svc/vpn-services/vpn-service"+
            "/application/app-id";
          }
          description
            "Identity for application";
        }
        leaf policy-id {
          type leafref {
            path "/sdwan-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
        "Grouping for flow definition.";
  }

  grouping application-criteria {
    list ac {
      key "name";
      ordered-by user;
      leaf name {
        type string;
        description
          "A description identifying application classification
```

```
             criteria.";
          }
        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.";
    }
    leaf topology {
      type identityref {
        base vpn-topology;
      }
      description
        "vpn topology: hub-and-spoke or any-to-any";
    }
    container performance-objective {
      leaf start-time {
        type yang:date-and-time;
        description
          "start-time indicates 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.";
      }
      description
        "The performance objective.";
    }
    container reserved-prefixes {
      leaf-list prefix {
        type inet:ip-prefix;
        description
          "ip prefix reserved for SP management purpose.";
      }
      description
        "ip prefix list reserved for SP management purpose.";
    }
    list application {
      key "app-id";
      leaf app-id {
        type svc-id;
        description
          "application name";
      }
      uses application-criteria;
      description
        "list 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 "../../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";
        }
        container policy-package {
          leaf encryption {
            type enumeration {
              enum yes {
                description
                  "Indicates whether or not the application flow requires
                   to send over encrypted overlay tunnel.";
              }
              enum either {
                description
                  " Either means this policy is not applied";
              }
            }
            description
              "Indicates whether or not the application flow requires
               encryption.";
          }
          leaf public-private {
            type enumeration {
              enum private-only {
                description
                  "The private WAN underlay is specified.";
              }
              enum either {
                description
                  "Both public WAN or private WAN could be used";
              }
            }
            description
              "Indicates whether the Application Flow can traverse
               Public or Private Underlay Connectivity Services
               (or both).Either means this policy is not applied.";
          }
```

```
      leaf local-breakout {
        type boolean;
        description
          "indicates whether the Application Flow should be
           routed directly to the Internet using Local Internet
           Breakout.It can have values Yes and No.";
      }
      leaf billing-method {
        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.";
      }
      leaf backup-path {
        type enumeration {
          enum yes {
            description
              "Only the primary tunnel overlay could be used for
               the traffic.";
          }
          enum no {
            description
              "Either the primary or backup overlay tunnel could be
               used for the traffic.";
          }
        }
        description
          "overlay connection as Primary or both Primary and
           Backup.";
      }
      container bandwidth {
        leaf commit {
          type uint32;
          description
            "CIR";
        }
        leaf max {
          type uint32;
```

```
          description
            "max speed ";
        }
        description
          "Container for the bandwidth policy";
      }
      description
        "Container for policy package";
    }
    description
      "List for policy";
  }
  list endpoints {
    key "endpoint-id";
    uses vpn-endpoint;
    description
      "List of endpoints.";
  }
  description
    "Grouping 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;
                  mandatory true;
                  description
                    "VLAN identifier.";
                }
                description
                  "Tagged interface.";
              }
              container priority-tagged {
                leaf tag-type {
                  type identityref {
                    base tag-type;
                  }
                  default "c-vlan";
                  description
                    "TAG type.By default, the TAG type is
                     Customer-VLAN tag.";
                }
                description
                  "Priority tagged.";
              }
              description
                "Container for tagged Interface.";
            }
            leaf l2-mtu {
```

```
         type uint32;
         units "bytes";
         description
           " L2 Maximum Frame Size MUST be an integer number of bytes
            >= 1522MTU.";
       }
       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.";
             }
             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: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
          "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.";
    }
    description
```

```
      "This grouping defines IP connection parameters.";
  }

container sdwan-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";
      }
      list device {
        key "name";
        leaf name {
          type string;
          description
            "Device Name";
        }
        leaf type {
          type identityref {
            base device-type;
          }
          description
            "Device Type: virtual or physical CE";
        }
        description
          "List for device";
      }
      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";
           }
           list wan-access {
             key "name";
             leaf name {
               type string;
               description
                 "wan access link name";
             }
             leaf access-type {
               type identityref {
                 base access-type;
               }
               description
                 "Access type: Internet, private VPN or cellular";
             }
             leaf access-provider {
               type string;
               description
                 "Specifies the name of provider";
             }
             container bandwidth {
               leaf input-bandwidth {
                 type uint64;
                 description
                   "input bandwidth";
               }
               leaf output-bandwidth {
                 type uint64;
                 description
                   "output bandwidth";
               }
               description
                 "Container for bandwidth";
             }
             uses site-l2-technology;
             uses site-ip-connection;
             description
               "container for wan access";
           }
           description
             "List for site";
         }
         description
           "Container for sites";
       }
       description
```

```
      "Top-level container for the SD-WAN services.";
  }
}
```

<CODE ENDS>

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
   [RFC8446].

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

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

8. IANA Considerations

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

              URI: urn:ietf:params:xml:ns:yang:ietf-sdwan-svc
              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-svc
              Namespace: urn:ietf:params:xml:ns:yang:ietf-sdwan-svc
              Prefix: sdwan-svc
              Reference: RFC xxxx

9. Appendix 1: Terminology Mapping between MEF SD-WAN Service
    Attributes and IETF SD-WAN model

   SD-WAN Service Attributes and Services [MEF70-Draft-R1], defines the
   SD-WAN service attributes and services for SD-WAN service delivery.
   These service attributes can be used for communication between
   subscribers and services to deliver SD-WAN services while this draft
   defines a YANG data model for SD-WAN service delivery communicated
   between customer and service provider. The purpose of both work is
   very similar.

   The below table shows the terminology mapping. The YANG model
   retains most parameter definition name but adjusts some of the
   structure to reserve space for future augmentation. For example, the
   model defines "vpn-service" and "lan-access" as a list, which can
   accommodate the case where the current MEF service attribute
   restricts only one VPN per customer and one LAN access and future
   extension to multiple VPN or LAN accesses per customer.

IETF SD-WAN Service model	MEF70 R1 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 link Attributes
wan-access	TBD(Underlay connectivity)

10. Appendix 2: IETF OSE model vs IETF SD-WAN model

   SD-WAN OSE service delivery model [I-D.wood-rtgwg-sdwan-ose-yang]
   defines two SD-WAN OSE Open SD-WAN Exchange (OSE) service YANG
   modules to enable the orchestrator in the enterprise network to
   implement SD-WAN inter-domain reachability and connectivity services
   and application aware traffic steering services. Although the OSE
   YANG model is also a service model instead of being a device model,
   this model is mainly used for interoperability between multiple SD-
   WAN domains and service consistency. The differences are shown as
   follows:

```
+-------------------------------+------------------------------+
| IETF OSE service model        | IETF SD-WAN Service model    |
+-------------------------------+------------------------------+
| Domain SD-WAN controller facing | customer-facing            |
|                               |                              |
+-------------------------------+------------------------------+
| Inter OSE GW connectivity service|unaware of SD-WAN domain in |
|                               |one SP network                |
|        Inter SD-WAN domain    |Inter-SD-WAN Service Provider |
|                               |TBD                           |
+-------------------------------+------------------------------+
| SLA aware dynamic Path selection |static Primary/Backup selection|
+-------------------------------+------------------------------+
```

For the SLA based dynamic path selection policy, the OSE service
model uses a similar application classification criteria, but at the
same time it will collect the relevant status of the traffic SLA
profiles and, based on the measurements calculated from the collected
information, the primary or secondary path will be selected.

```
 +--primary-backup
     +--rw path-values
        +--rw sla-values
           +--rw latency?          uint32
           +--rw jitter?           uint32
           +--rw packet-loss-rate? uint32
```

11. Acknowledgments

This work has benefited from the discussions of with Jack
Pugaczewski, Larry S Samberg, and Pascal Menezes from MEF community.

12. Contributors

The authors would like to thank Zitao Wang for his major
contributions to the initial modelling.

13. References

13.1. Normative References

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
            Requirement Levels", BCP 14, RFC 2119,
            DOI 10.17487/RFC2119, March 1997,
            <https://www.rfc-editor.org/info/rfc2119>.

13.2. Informative References

   [I-D.wood-rtgwg-sdwan-ose-yang]
              Wood, S., Bo, W., Wu, Q., and C. Menezes, "YANG Data Model
              for SD-WAN OSE service delivery", draft-wood-rtgwg-sdwan-
              ose-yang-00 (work in progress), March 2019.

   [MEF51.1] MEF, Ed., "Operator Ethernet Service Definition", December
              2018, <https://wiki.mef.net/display/CESG/
              MEF+51.1+-+OVC+Services>.

   [MEF70-Draft-R1]
              MEF, Ed., "SD-WAN Service Attributes and Services", May
              2019, <https://www.mef.net/Assets/Draft-Standards/
              MEF_70_Draft_(R1).pdf>.

   [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P.
              Traina, "Generic Routing Encapsulation (GRE)", RFC 2784,
              DOI 10.17487/RFC2784, March 2000,
              <https://www.rfc-editor.org/info/rfc2784>.

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

   [RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual
              Private Network (VPN) Terminology", RFC 4026,
              DOI 10.17487/RFC4026, March 2005,
              <https://www.rfc-editor.org/info/rfc4026>.

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

   [RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer
              2 Virtual Private Networks (L2VPNs)", RFC 4664,
              DOI 10.17487/RFC4664, September 2006,
              <https://www.rfc-editor.org/info/rfc4664>.

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

   [RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and
              Internet Key Exchange (IKE) Document Roadmap", RFC 6071,
              DOI 10.17487/RFC6071, February 2011,
              <https://www.rfc-editor.org/info/rfc6071>.

   [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
              and A. Bierman, Ed., "Network Configuration Protocol
              (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,
              <https://www.rfc-editor.org/info/rfc6241>.

   [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure
              Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011,
              <https://www.rfc-editor.org/info/rfc6242>.

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

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

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

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

   [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol
              Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018,
              <https://www.rfc-editor.org/info/rfc8446>.

   [RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG
              Data Model for Layer 2 Virtual Private Network (L2VPN)
              Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October
              2018, <https://www.rfc-editor.org/info/rfc8466>.

Authors' Addresses

   Qiong Sun
   China Telecom
   Beijing
   China

   Email: sunqiong.bri@chinatelecom.cn

   Honglei Xu
   China Telecom
   Beijing
   China

   Email: xuhl.bri@chinatelecom.cn

   Bo Wu (editor)
   Huawei
   Nanjing
   China

   Email: lana.wubo@huawei.com

   Qin Wu (editor)
   Huawei
   Nanjing
   China

   Email: bill.wu@huawei.com

   Charles Eckel (editor)
   Cisco Systems
   170 W. Tasman Drive
   San Jose, CA
   United States

   Email: eckelcu@cisco.com