Abstract

This document defines a YANG data model that can be used by a network operator to configure a VPN service at one layer interconnecting VPN service at another layer and manage how a hybrid VPN service is engineered in the network [RFC8309]. This model is intended to be instantiated at the management system to deliver the overall service. It is not a configuration model to be used directly on network elements. This model provides an abstracted view of VPN service configuration components at different layer. It is up to a management system to take this as an input and generate specific configurations models to configure the different network elements to deliver the service. How configuration of network elements is done is out of scope of the document.

Status of This Memo

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

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

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

This Internet-Draft will expire on March 31, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

BGP/MPLS IP Virtual Private Networks (IP VPNs) [RFC4364] have been widely deployed to provide network based Layer 3 VPNs solutions. As MPLS-based Layer 2 services grow in demand, many mobile SPs are interested to extend their conventional L2VPN at the X1 interface in the metro access network into IP VPN capabilities at the S1 interface between access network and core network to provide end-to-end native BGP IP VPN services to their enterprise customers. Another benefit
is to enable the sharing of a provider’s core network infrastructure between IP and Layer 2 VPN services.

This document defines a YANG data model that can be used by a network operator to configure a VPN across multi-domain environment with a VPN service at one layer interconnecting a VPN service at another layer and manage how a hybrid VPN service is engineered in the network [RFC8309]. This model is intended to be instantiated at the management system to deliver the overall service. It is not a configuration model to be used directly on network elements. This model provides an abstracted view of VPN service configuration components at different layer. It is up to a management system to take this as an input and generate specific configurations models to configure the different network elements to deliver the service. How configuration of network elements is done is out of scope of the document.

1.1. Terminology

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

- client
- server
- configuration data
- state data

The following terms are defined in [RFC7950] and are not redefined here:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC7950].

1.1.1. Requirements Language

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

1.2. Tree diagram

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Definitions

This document uses the following terms:

Service Provider (SP): The organization (usually a commercial undertaking) responsible for operating the network that offers VPN services to clients and customers.

Customer Edge (CE) Device: Equipment that is dedicated to a particular customer and is directly connected to one or more PE devices via attachment circuits. A CE is usually located at the customer premises, and is usually dedicated to a single VPN, although it may support multiple VPNs if each one has separate attachment circuits. The CE devices can be routers, bridges, switches, or hosts.

Provider Edge (PE) Device: Equipment managed by the SP that can support multiple VPNs for different customers, and is directly connected to one or more CE devices via attachment circuits. A PE is usually located at an SP point of presence (PoP) and is managed by the SP.

3. The Composed VPN Service Model

A Composed VPN service is a collection of VPN sites that are authorized to exchange traffic between each other over a shared infrastructure of a common L3VPN technology. This Composed VPN service model provides a common understanding of how the corresponding composed VPN service is to be deployed in an end to end manner over the shared infrastructure.

This document presents the Composed VPN Service Delivery Model using the YANG data modeling language [RFC7950] as a formal language that is both human-readable and parsable by software for use with protocols such as NETCONF [RFC6241] and RESTCONF [RFC8040].
3.1. VPN Service Types

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

- Layer 2 VPN Service
- Layer 3 VPN Service
- VXLAN Service

3.2. Composed VPN Physical Network Topology

NodeBs of the site 1,2,3,4 in the access metro network are one end of the Layer 2 VPN and communicate with each other via X2 interface. sGW/MME of site 5,6 in the core network are another end of Layer 3 VPN and communicate NodeB in site 1,2,3,4 via S1 interface in the access metro network. The Router PE at the edge of the access metro network is performing the VPN stitching between the Layer 2 VPN and the Layer 3 VPN using the technology such as bridging or other interconnection technology. The Router PE uses the logical tunnel interface (lt interface) configured with different logical interface units applied under two different VPN instances. Therefore the end to end VPN connection spanning across the metro access network and the IP core network can be established.
4. Service Model Usage

L2VPN interconnection with L3VPN in Mobile Backhaul Network

L2VPN interconnection with L3VPN in Mobile Backhaul Network
The idea of the composed VPN model is to decompose end to end VPN service across multiple-domain environment into segmented tunnel or segmented VPN service in each domain. A typical scenario would be to use composed VPN model spanning across multi-domain as an input for an orchestration layer that will be responsible for translating it to an orchestrated per domain configuration of network elements that will be part of the service. The per domain configuration of network elements will be defined as segmented VPN model in each domain. The configuration of network elements can be done via the CLI, NETCONF/RESTCONF [RFC6241][RFC8040] coupled with YANG data models of a specific configuration (BGP, VRF, BFD, etc.), or some other technique, as preferred by the operator. Another scenario is to use customer facing model such as L3SM service model as an input for an orchestration layer that will be responsible for translating it to the composed VPN model and the composed VPN model can be further broken down into per domain segmented VPN model in each domain.

The usage of this composed VPN model is not limited to this example; it can be used by any component of the management system but not directly by network elements.
5. Design of the Data Model

The YANG data model for the composed VPN has been split into two YANG modules. The first module, "ietf-composed-vpn-svc", defines global parameters and the essential components of a composed VPN that are used to provide end to end connectivity spanning across multiple domains. The second module "ietf-segmented-vpn" defines per domain segmented vpn parameters and associated access point list parameters that are used to connect to the peer device or domain. The segmented vpn list defined in the second module "ietf-segmented-vpn" also provides basic component for the first module "ietf-composed-vpn-svc".

The global parameters under composed-vpns container contain generic information about the VPN service such as vpn-id, customer-name, service-topology and operational state. The "vpn-id" provided in the composed-vpn list refers to an internal reference for this composed VPN service, while the customer name refers to a more-explicit reference to the customer. This identifier is purely internal to the organization responsible for the composed VPN service. vpn-topology describes the type of VPN service topology is required for configuration. Our proposed model supports any-to-any, Hub and Spoke.Operation state includes admin-state, oper-state, sync-State and start-time.

Two essential components of a composed VPN are segmented vpns list and access points list. A composed-vpn is composed of at least one access points and at least one segmented vpns. A segmented vpn under "segmented-vpn" list is composed of at least one access point. The access point parameters under comosed-vpn level is used to describe end to end connectivity stitching with one or multiple access connectivities in each segmented VPN while the access point parameters under segmented VPN level is used to describe one or multiple access connectivities pertaining to each segment VPN.

Figure 1 shows abridged views of the hierarchies.
6. Basic Building Blocks

This section describes the essential components of the composed VPN data model.

6.1. Access Points

The access points are basic element information in a composed VPN and used to describe any network access connectivity across domains or within a domain. The access points list models a list of access points including the access or attachment information for the remote peer. The access point provided by the remote peer (e.g., PE) at the composed VPN level are used to describe the network access connectivity across domains. The access point provided by the remote peer (e.g., PE) at the segmented VPN level are used to describe network access connectivity within a domain. The access point uses working layer and corresponding layer information to describe the different configurations in composed VPN level and the segmented VPN level.

As can be seen from Figure 1, the access points list further introduces several generic components of a composed VPN: termination point basic information, peer CE node information, QoS and routing protocols. Section 6.1.1, Section 6.1.2 and Section 6.1.3 describes these components in more detail.

```
+--rw composed-vpns
  +--rw composed-vpn* [vpn-id]
    +--rw vpn-id            yang:uuid
    +--rw vpn-name?         string
    +--rw customer-name?    yang:uuid
    +--rw topo?             svpn:topology
    +--rw service-type?     svpn:service-type
    +--rw technology?       svpn:tunnel-type
    +--rw admin-state?      svpn:admin-state
    +--ro oper-State?       svpn:oper-state
    +--ro sync-state?       svpn:sync-state
    +--rw start-time?       yang:date-and-time
  +--rw seg-vpns* [index]
    |    ...
    +--rw access-points* [tp-id]
      +--rw tp-basic
      +--rw peer-ce-node
      +--rw route-protocol* [type]
```

Figure 1: Figure 1
6.1.1. Termination Point Basic Information

The tp-basic list describes the basic information that is used to express attachment point information (e.g., PE information) and QoS requirements (see section 6.1.1.1 for details) of the network access connectivity from the Termination Point (TP) of view. That means the information described here is relative static, no matter which two pair of peer TPs are going to connect.
The type-Spec list describes the layered information on the TP, such as ethernet layer information, or the IP layer and VXLAN information if any higher layer protocol is enabled.

6.1.1.1. QoS

This model supports two kinds of QoS requirements as described in the section 7 and section 8:

TP based QoS: Describes the QoS attributes on a termination point. For example, the CAR (committed access rate) definition on the inbound or outbound ports.

Flow based QoS: Describes the QoS attributes on a service flow. This enables the fine grained QoS control with the capability of identifying the service flow.

Both are applicable to network access connectivity between any two peers within domain or across domain.

For TP based QoS, this model defines the following QoS attributes:

- "qos-config-type": Specify QoS configuration type. This model support the following setting: standard and custom.
- "qos-sub-type": Specify QoS detailed configuration type. This model support the following settings: car, diffserv, diffservdomain, QoS Profile.
- "in-profile-id": Identification of the QoS Profile to be used for downstream traffic. Local administration meaning.
- "out-profile-id": Identification of the QoS Profile to be used for upstream traffic. Local administration meaning.
- "in-tp-car": Describe service bandwidth configurations for downstream traffic.
- "out-tp-car": Describe service bandwidth configurations for downstream traffic.

For Flow based QoS, this model defines the following QoS attributes:

- "qos-config-type": Specify QoS configuration type. This model support the following setting: standard and custom.
6.1.2. Peer CE Node

The peer-ce-node describes CE node information associated with access point including CE node information, TP information, address and location. The peer CE node can be used together with the node-id and node-addr under access-point list to identify the source endpoint and destination endpoint of network access connectivity between any two routers (e.g., PE or ASBR) at the edge of each domain.

6.1.3. Routing Protocols

The "routing-protocol" list defines which routing protocol must be activated between any two routers (e.g., PE or ASBR) at the edge of each domain. The current model supports the following settings: bgp, rip, ospf, static. In addition, the "routing-protocol" list in this model can be augmented with any possible routing protocols. The BGP and static routing list are examples to show how these two widely used routing protocols are described.

6.2. Segmented VPN

As a large network grows, it might become desirable to partition the network into multiple domains or segments. The seg-vpn list describes a list of Segmented VPN information from the segment point of view. i.e., specify how it communicates with peered devices outside this Segmented VPN. The segmented VPN information is composed of basic VPN information and a list of access points. The set of access points are outgoing interfaces that customer sites or other segmented VPNs can attach. The Segmented VPN could be a layer 2 VPN, or layer 3 VPN, or even a termination point. The seg-vpn list can be used as key component for both "ietf-composed-vpn-svc" and "ietf-segmented-vpn".
+--rw segment-vpns
  +--rw segment-vpn* [index]
    +--rw index                uint32
    +--rw protect-role?        protection-role
    +--rw vpn-info
      +--rw (vpn-type)?
        +--:(wanvpn)
          +--rw vpn
            +--rw vpn-id?              yang:uuid
            +--rw vpn-name?            string
            +--rw topo?                Topology
            +--rw service-type?        service-type
            +--rw technology?          tunnel-type
            +--rw admin-state?          admin-state
            +--ro oper-state?           oper-state
            +--ro sync-state?           sync-state
            +--rw access-point* [tp-id]

7. Composed VPN YANG Module

<CODE BEGINS> file "ietf-composed-vpn-svc.yang"
module ietf-composed-vpn-svc {
  prefix cvpn ;
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-segmented-vpn {
    prefix svpn;
  }
  organization "IETF OPSAWG Working Group";
  contact "WG Web: <https://datatracker.ietf.org/wg/opsawg>
  WG List: <mailto:netmod@ietf.org>
  Editor: Roni Even
    <mailto:roni.even@huawei.com>
  Qin Wu
    <mailto:bill.wu@huawei.com>
  Ying Cheng
    <mailto:chengying10@chinaunicom.cn>";
  description "ietf-composed-vpn";
  revision 2018-08-21 {
    reference "draft-evenwu-opsawg-yang-composed-vpn-00"
  }
}

grouping vpn-basic {
description "VPNBasicInfo Grouping.";
leaf topo {
    type svpn:topology;
    description "current support for full-mesh and
    point_to_multipoint(hub-spoke), others is reserved for
    future extensions." ;
}
leaf service-type {
    type svpn:service-type;
    description "current support for mpls l3vpn/vxlan/L2VPN
    overlay, others is reserved for future extensions." ;
}
leaf technology {
    type svpn:tunnel-type;
    description "mpls|vxlan overlay l3vpn|eth over sdh|nop";
}
leaf admin-state {
    type svpn:admin-state;
    description "administrative status." ;
}
leaf oper-State {
    type svpn:oper-state;
    config false;
    description "Operational status." ;
}
leaf sync-state {
    type svpn:sync-state;
    config false;
    description "Sync status." ;
}
leaf start-time {
    type yang:date-and-time;
    description "Service lifecycle: request for service start
    time." ;
}
}

container composed-vpns{
    description "";
    list composed-vpn {
        key "vpn-id";
        description "List for composed VPNs.";
        uses composedvpn;
    }
}

grouping composedvpn {
    description "ComposedVPN Grouping.";
}
leaf vpn-id {
  type yang:uuid;
  description "Composed VPN identifier." ;
}
leaf vpn-name {
  type string {length "0..200";}
  description "Composed VPN Name. Local administration meaning" ;
}
leaf customer-name {
  type yang:uuid;
  description
  "Name of the customer that actually uses the VPN service.
   In the case that any intermediary (e.g., Tier-2 provider
   or partner) sells the VPN service to their end user
   on behalf of the original service provider (e.g., Tier-1
   provider), the original service provider may require the
   customer name to provide smooth activation/commissioning
   and operation for the service." ;
}
uses vpn-basic;
list seg-vpns {
  key "index";
  description "SegVpn list ";
  uses svpn:segment-vpn;
}
list access-points {
  key "tp-id";
  description "TP list of the access links which associated
   with CE and PE";
  uses svpn:termination-point;
}

8. Segment VPN YANG Module

<CODE BEGINS> file "ietf-segmented-vpn.yang"
module ietf-segmented-vpn {
  prefix svpn;
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

organization "IETF OPSAWG Working Group";
contact "
WG Web:  <https://datatracker.ietf.org/wg/opsawg>
WG List:  <mailto:netmod@ietf.org>
Editor:   Roni Even
          <mailto:roni.even@huawei.com>
          Qin Wu
          <mailto:bill.wu@huawei.com>
          Cheng Ying
          <mailto:chengying10@chinaunicom.cn>">

description
 "This YANG module defines a generic service configuration
 model for Composed VPNs. This model is common across all
 vendor implementations."
revision 2018-08-21 {
    reference "draft-opsawg-evenwu-yang-composed-vpn-00";
}
typedef edge-role {
    type enumeration {
        enum nop {
            description "nop";
        }
        enum pe {
            description "PE";
        }
        enum p {
            description "P";
        }
        enum uni {
            description "UNI";
        }
        enum nni {
            description "NNI";
        }
        enum asbtp {
            description "AsbTP";
        }
    }
    description "Edge Point Role.";
}
typedef topo-role {
    type enumeration {
        enum hub {
            description "hub";
        }
        enum spoke {
            description "spoke";
        }
    }
    description "Topo Role.";
}
enum other {
    description "other";
}

description "Topo Node Role.";

typedef protection-role {
    type enumeration {
        enum nop{
            description "NOP";
        }
        enum main{
            description "MAIN";
        }
    }
    description "Protection Role.";
}

typedef qos-config-type {
    type enumeration {
        enum nop{
            description "nop.";
        }
        enum template{
            description "standard.";
        }
        enum agile{
            description "custom.";
        }
    }
    description "Qos Config Type.";
}

typedef qos-sub-type {
    type enumeration {
        enum nop{
            description "nop";
        }
        enum car{
            description "CAR";
        }
        enum qosProfile{
            description "Qos Profile";
        }
        enum diffservdomain{
            description "diffServDomain";
        }
        enum diffserv{
            description "diffServ";
        }
    }
    description "Qos Sub Type.";
}
typedef tp-type{
  type enumeration {
    enum nop {
      description "nop";
    }
    enum ptp {
      description "PTP";
    }
    enum ctp {
      description "CTP";
    }
    enum trunk {
      description "TRUNK";
    }
    enum loopback {
      description "LoopBack";
    }
    enum tppool {
      description "TPPool";
    }
  }
  description "Tp Type.";
}

typedef layer-rate{
  type enumeration {
    enum lr-unknow {
      description "Layer Rate UNKNOW.";
    }
    enum lr-ip {
      description "Layer Rate IP.";
    }
    enum lr-eth {
      description "Layer Rate Ethernet.";
    }
    enum lr_vxlan {
      description "Layer Rate VXLAN.";
    }
  }
  description "Layer Rate.";
}

typedef admin-state {
  type enumeration {
    enum active {
      description "Active status";
    }
    description "Administrative State.";
  }
}
enum inactive {
    description "Inactive status";
}
enum partial {
    description "Partial status";
}

description "Admin State.";

typedef oper-state {
    type enumeration {
        enum up {
            description "Up status";
        }
        enum down {
            description "Down status";
        }
        enum degrade {
            description "Degrade status";
        }
    }
    description "Operational Status.";
}

typedef sync-state {
    type enumeration {
        enum sync {
            description "Sync status";
        }
        enum out-sync {
            description "Out sync status";
        }
    }
    description "Sync Status";
}

typedef eth-encap-type {
    type enumeration {
        enum default {
            description "DEFAULT";
        }
        enum dot1q {
            description "DOT1Q";
        }
        enum qinq {
            description "QINQ";
        }
        enum untag {
            description "UNTAG";
        }
    }
}
typedef protocol-type {
  type enumeration {
    enum static {
      description "Static Routing";
    }
    enum bgp {
      description "bgp";
    }
    enum rip {
      description "rip";
    }
    enum ospf {
      description "ospf";
    }
    enum isis {
      description "isis";
    }
  }
  description "Routing Protocol Type";
}

typedef tunnel-type {
  type enumeration {
    enum NOP{
      description "NOP";
    }
    enum MPLS{
      description "MPLS";
    }
    enum MPLS-TP{
      description "MPLS-TP";
    }
    enum MPLS-SR {
      description "MPLS Segment Routing";
    }
    enum SRv6 {
      description "SRv6";
    }
  }
  description "VPN Tunnel Type.";
}

typedef service-type {
  type enumeration {
    enum l3vpn {

typedef topology {
    type enumeration {
        enum any-to-any {
            description "any to any";
        }
        enum hub-spoke {
            description "hub and spoke VPN topology.";
        }
        enum hub-spoke-disjoint {
            description "Hub and spoke VPN topology where Hubs cannot communicate with each other ";
        }
        enum unknown {
            description "unknown.";
        }
    }
    description "Topology.";
}

typedef ethernet-action {
    type enumeration {
        enum nop {
            description "nop";
        }
        enum untag {
            description "UNTAG";
        }
        enum stacking {
            description "STACKING";
        }
    }
    description "Ethernet Action.";
}

typedef color-type {
    type enumeration {
        enum green {
            description "green";
        }
    }
}
enum yellow{
    description "yellow";
}
enum red{
    description "red";
}
enum all{
    description "all";
}

description "Color Type."
}
typedef action-type {
    type enumeration {
        enum nop{
            description "nop";
        }
        enum bandwidth{
            description "bandwidth";
        }
        enum pass{
            description "pass";
        }
        enum discard{
            description "discard";
        }
        enum remark{
            description "remark";
        }
        enum redirect{
            description "redirect";
        }
        enum recolor{
            description "recolor";
        }
        enum addRt{
            description "addRt";
        }
    }
    description "Action Type";
}

//typedef PWTagMode {
    type enumeration {
        enum raw{

enum tagged{
  description "TAGGED";
}

description "PWTagMode";
}

grouping QinQVlan {
  description "QinQVlan Grouping.";
  leaf-list cvlan {
    type uint64;
    description "cvlan List.";
  }
  leaf svlan {
    type uint64;
    description "svlan.";
  }
}

grouping Dot1QVlan {
  description "Dot1QVlan Grouping.";
  leaf-list dot1q {
    type uint64;
    description "dot1q Vlan List";
  }
}

//TpTypeSpec

grouping tp-type-spec{
  description "Tp Type Spec Grouping.";
  leaf layer-rate {
    type layer-rate;
    description "layer Rate";
  }
  choice spec-value {
    description "Spec Value";
    case lr-eth {
      container eth {
        description "ethernetSpec";
        uses ethernet-spec;
      }
    }
    case lr-ip {
      container ip {
        description "ipSpec";
        uses IpSpec;
      }
    }
    case lr-pw {

container pw {
    description "PwSpec";
    uses PwSpec;
}

case lr-vxlan {
    container vxlan {
        description "vxlanSpec";
        uses VxlanSpec;
    }
}

grouping FlowServices {
    description "FlowServices Grouping.";
    leaf qos-config-type {
        type qos-config-type;
        description "qos Config Type.";
    }
    leaf qos-sub-type {
        type qos-sub-type;
        description "qosDetailType";
    }
    leaf in-template-id {
        type yang:uuid;
        description "in Flow Qos Template ID.";
    }
    leaf out-template-id {
        type yang:uuid;
        description "out Flow Qos Template ID.";
    }
    list flow-service {
        key class-id;
        description "default in flow and behaviors";
        uses FlowAndBehavior;
    }
}

grouping TPQosNode {
    description "TPQosNode Grouping.";
    leaf qos-config-type {
        type qos-config-type;
        description "qos Config Type.";
    }
    leaf qos-sub-type {
        type qos-sub-type;
        description "qos Detail Type";
    }
    leaf in-profile-id {
type yang:uuid;
description "inQosProfileId";
}
leaf out-profile-id {
  type yang:uuid;
  description "outQosProfileId";
}
list in-tp-car {
  key index;
  uses FlowBehavior;
  description "inTpCar";
}
list out-tp-car {
  key index;
  uses FlowBehavior;
  description "outTpCar";
}

//CE spec

grouping CeTp {
  description "CeTp Grouping.";
  leaf ce-id {
    type yang:uuid;
    description "Site router ID";
  }
  leaf ce-node-id {
    type yang:uuid;
    description "directly connected NE node ID, only valid in asbr";
  }
  leaf ce-tp-id {
    type yang:uuid;
    description "ce Directly connected TP id, only valid in asbr";
  }
  leaf ce-address {
    type inet:ip-address;
    description "ceIfmasterIp";
  }
  leaf location {
    type string {length "0..400"};
    description "CE device location";
  }
}

//TPBasicInfo

grouping TPBasicInfo {
  description "TPBasicInfo Grouping.";
  leaf tp-id {

type yang:uuid;
  description "An identifier for termination point on a node.";
}

leaf tp-name {
  type string {length "0..200"};
  description "The termination point Name on a node. It conforms to name rule defined in system. Example FE0/0/1, GE1/2/1.1, Eth-Trunk1.1, etc";
}

leaf node-id {
  type yang:uuid;
  description "Identifier for a node.";
}

leaf edge-role {
  type edge-role;
  description "edge role for TP, for example:UNI/NNI";
}

leaf topo-role {
  type topo-role;
  description "hub/spoke role, etc";
}

leaf tp-type {
  type tp-type;
  description "Type";
}

leaf working-layer {
  type layer-rate;
  description "working layer";
}

list type-spec {
  key "layer-rate";
  uses tp-type-spec;
  description "typeSpecList";
}

container tp-qos-node {
  description "tp Qos Node."
  uses TPQosNode;
}

container flow-services {
  description "flow services in one TP."
  uses FlowServices;
}

grouping RouteProtocolSpec {
  description "Routing Protocol Spec Grouping."
  leaf type {
    type protocol-type;
    description "Protocol type";
  }
}
choice para {
  description "para" ;
  case static {
    list static {
      key "index";
      uses StaticRouteItem;
      description "staticRouteItems" ;
    }
  }
  case bgp {
    list bgp {
      key "index";
      uses BGPProtocolItem;
      description "bgpProtocols" ;
    }
  }
}
grouping BGPProtocolItem {
  description "BGPProtocolItem Grouping.";
  leaf index {
    type uint32;
    description "index of BGP protocol item";
  }
  leaf as-no {
    type uint64;
    description "Peer AS Number.";
  }
  leaf max-prefix {
    type int32;
    description "maximum number limit of prefixes.";
  }
  leaf max-prefix-alarm {
    type uint32;
    description "alarm threshold of BGP rout";
  }
  leaf peer-address {
    type inet:ip-address;
    description "peerIp";
  }
}
grouping StaticRouteItem {
  description "StaticRouteItem Grouping.";
  leaf index {
    type uint32;
    description "static item index";
  }
leaf dest-cidr {
  type string;
  description "address prefix specifying the set of
  destination addresses for which the route may be
  used.";
}
leaf egress-tp {
  type yang:uuid;
  description "egress tp";
}
leaf route-preference {
  type string;
  description "route priority. Ordinary, work route have
  higher priority.";
}
leaf next-hop {
  type inet:ip-address;
  description "Determines the outgoing interface and/or
  next-hop address(es), or a special operation to be
  performed on a packet.";
}
}
grouping ethernet-spec {
  description "Ethernet Spec Grouping.";
  leaf access-type {
    type eth-encap-type;
    description "access frame type";
  }
  choice accessVlanValue {
    description "accessVlanValue";
    case qinq {
      container qinq {
        description "qinqVlan";
        uses QinQVlan;
      }
    }
    case dot1q {
      container dot1q {
        description "dot1qVlan";
        uses Dot1QVlan;
      }
    }
  }
  leaf vlan-action {
    type ethernet-action;
    description "specify the action when the vlan is matched";
  }
  leaf action {
type string{length "0..100";}  
    description "specify the action value.";
}

grouping PwSpec {
    description "PwSpec Grouping.";
    leaf control-word {
        type boolean;
        default false;
        description "control Word.";
    }
    leaf vlan-action {
        type PWTagMode;
        description "pw Vlan Action.";
    }
}

grouping IpSpec {
    description "IpSpec Grouping.";
    leaf ip-address {
        type inet:ip-address;
        description "master IP address";
    }
    leaf mtu {
        type uint64;
        description "mtu for ip layer,scope:46~9600";
    }
}

grouping VxlanSpec {
    description "VxlanSpec Grouping.";
    leaf vni {
        type uint32;
        description "vni";
    }
    leaf vtep-ip {
        type inet:ip-address;
        description "vtep ip";
    }
}

grouping FlowAndBehavior {
    description "FlowAndBehavior Grouping.";
    leaf class-id {
        type yang:uuid;
        description "flowClassifierId";
    }
    list flow-behavior {
        key index;
        uses FlowBehavior;
        description "flowBehaviors";
    }
}
grouping FlowBehavior {
    description "FlowAndBehavior Grouping.";
    leaf index {
        type uint32;
        description "index";
    }
    leaf color-type {
        type color-type;
        description "Color Type.";
    }
    leaf action-type {
        type action-type;
        description "action Type";
    }
    leaf action {
        type string;
        description "action";
    }
}

grouping VPNBasicInfo {
    description "VPNBasicInfo Grouping.";
    leaf topo {
        type topology;
        description "current support for full-mesh and
        hub-spoke, others is reserved for
        future extensions.";
    }
    leaf service-type {
        type service-type;
        description "current support for mpls l3vpn/vxlan/L2VPN
        overlay, others is reserved for future extensions.";
    }
    leaf technology {
        type tunnel-type;
        description "mpls|vxlan overlay l3vpn|eth over sdh|nop";
    }
    leaf admin-state {
        type admin-state;
        description "administrative status.";
    }
    leaf oper-state {
        type oper-state;
        config false;
        description "Operational status.";
    }
    leaf sync-state {
type sync-state;
  config false;
  description "Sync status.";
}

grouping VPN {
  description "VPN Grouping.";
  leaf vpn-id {
    type yang:uuid;
    description "VPN Identifier.";
  }
  leaf vpn-name {
    type string {length "0..200"};
    description "Human-readable name for the VPN service.";
  }
  uses VPNBasicInfo;
  list access-point {
    key "tp-id";
    description "TP list of the access links which associated
    with CE and PE";
    uses termination-point;
  }
}

grouping termination-point {
  description "grouping for termination points.";
  leaf tp-id {
    type yang:uuid;
    description "An identifier for termination point on a node.";
  }
  container peer-ce-node {
    description "CE TP Information.";
    uses CeTp;
  }
  container tp-basic {
    description "Termination point basic info.";
    uses TPBasicInfo;
  }
  list route-protocol {
    key "type";
    description "route protocol spec.";
    uses RouteProtocolSpec;
  }
  leaf admin-state {
    type admin-state;
    description "administrative status.";
  }
  leaf oper-state {
    type oper-state;
  }

config false;
  description "Operational status." ;
}
}
grouping segment-vpn {
  description "SegmentVPN Grouping.";
  leaf index {
    type uint32;
    description "index of segment VPN in a composed VPN.";
  }
  leaf protect-role {
    type protection-role;
    description "The protection role of segment VPN, by
default it is set as nop role.";
  }
  container vpn-info {
    description "vpn information";
    choice vpn-type {
      description "vpn type.";
      case wan-vpn {
        container vpn {
          description "vpn.";
          uses VPN;
        }
      }
    }
  }
}
container segment-vpns {
  list segment-vpn {
    key "index";
    description "Segment Vpn list.";
    uses segment-vpn;
    description "Container for Segment VPN.";
  }
}

9. Service Model Usage Example

This section provides an example of how a management system can use
this model to configure an IP VPN service on network elements.
In this example, we want to achieve the provisioning of a end to end VPN service for three sites using a Hub-and-Spoke VPN service topology. The end to end VPN service is stitched by three segmented VPN, two are within intra-AS domain, one is within inter AS domain.

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

```xml
<?xml version="1.0"?>
<composed-vpns xmlns="urn:ietf:params:xml:ns:yang:ietf-composed-vpn-svc">
  <composed-vpn>
    <vpn-id>12456487</vpn-id>
    <topo>hub-spoke</topo>
    <service-type>hybrid-vpn</service-type>
    <seg-vpns>
      <index>1</index>
      <vpn-info>
        <vpn-id>111</vpn-id>
        <topo>hub-spoke</topo>
        <service-type>l2vpn</service-type>
        <access-point>
          <node-id>ASBR1</node-id>
          <peer-ce-node>
            <ce-node-id>PE1</ce-node-id>
          </peer-ce-node>
          <tp-basic>
            <topo-role>hub</topo-role>
            <flow-services>
              <in-template-id>TEMPLATE-A</in-template-id>
              <out-template-id>TEMPLATE-B</out-template-id>
            </flow-services>
          </tp-basic>
        </access-point>
      </vpn-info>
    </seg-vpns>
  </composed-vpn>
</composed-vpns>
```
<bgp>
  <as-no>AS1</as-no>
</bgp>

<routing-protocol>
</access-point>
</vpn-info>

<seg-vpns>
  <index>2</index>
  <vpn-info>
    <vpn-id>222</vpn-id>
    <topo>hub-spoke</topo>
    <service-type>l3vpn</service-type>
    <access-point>
      <node-id>ASBR2</node-id>
      <peer-ce-node>
        <ce-node-id>ASBR1</ce-node-id>
      </peer-ce-node>
      <tp-basic>
        <topo-role>hub</topo-role>
        <flow-serices>
          <in-template-id>TEMPLATE-B</in-template-id>
          <out-template-id>TEMPLATE-C</out-template-id>
        </flow-serices>
      </tp-basic>
    </access-point>
  </vpn-info>
</seg-vpns>

<seg-vpns>
  <index>3</index>
  <vpn-info>
    <vpn-id>333</vpn-id>
    <topo>hub-spoke</topo>
    <service-type>l2vpn</service-type>
    <access-point>
      <node-id>PE2</node-id>
      <peer-ce-node>
        <ce-node-id>ASBR2</ce-node-id>
      </peer-ce-node>
      <tp-basic>
        <topo-role>spoke</topo-role>
        <flow-serices>
          <in-template-id>TEMPLATE-B</in-template-id>
        </flow-serices>
      </tp-basic>
    </access-point>
  </vpn-info>
</seg-vpns>
10. Interaction with other YANG models

As expressed in Section 4, this composed VPN service model is intended to be instantiated in a management system and not directly on network elements.

The management system’s role will be to configure the network elements. The management system may be modular and distinguish the component instantiating the service model (let’s call it "service component") from the component responsible for network element configuration (let’s call it "configuration component"). The service is built from a combination of network elements and protocols configuration which also include various aspects of the underlying network infrastructure, including functions/devices and their subsystems, and relevant protocols operating at the link and network layers across multiple device. Therefore there will be a strong relationship between the abstracted view provided by this service model and the detailed configuration view that will be provided by specific configuration models for network elements.

The service component will take input from customer service model such as L3SM service model or composed VPN service model and translate it into segment VPN in each domain and then further break down the segment VPN into detailed configuration view that will be provided by specific configuration models for network elements.

11. 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:

- /composed-vpns/composed-vpn

  The entries in the list above include the whole composed vpn service configurations which the customer subscribes, and indirectly create or modify the PE,CE and ASBR device configurations. Unexpected changes to these entries could lead to service disruption and/or network misbehavior.

- /composed-vpns/composed-vpn/seg-vpns

  The entries in the list above include the access points configurations. As above, unexpected changes to these entries could lead to service disruption and/or network misbehavior.

- /composed-vpns/composed-vpn/access-points

  The entries in the list above include the access points configurations. As above, unexpected changes to these entries could lead to service disruption and/or network misbehavior.

12. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested to be made:
This document registers two YANG modules in the YANG Module Names registry [RFC6020].

```
Name: ietf-composite-vpn-svc
Prefix: composite-svc
Reference: RFC xxxx

Name: ietf-segmented-vpn
Prefix: segment-vpn
Reference: RFC xxxx
```

13. Normative References


Appendix A.  Acknowledges

Geng Liang, Congfeng Xie, Chen Rui, LiYa Zhang, Hui Deng contributed to an earlier version of [I-D.chen-opsawg-composite-vpn-dm]. We would like to thank the authors of that document on the operators’ view for the PE-based VPN service configuration for material that assisted in thinking about this document.

Authors’ Addresses

Roni Even
Huawei Technologies, Co., Ltd
Tel Aviv
Israel

Email: roni.even@huawei.com