Internet Engineering Task Force (IETF)
Request for Comments: 8049
Category: Standards Track
ISSN: 2070-1721

S. Litkowski
Orange Business Services
L. Tomotaki
Verizon
K. Ogaki
KDDI Corporation

February 2017

YANG Data Model for L3VPN Service Delivery

Abstract

This document defines a YANG data model that can be used for communication between customers and network operators and to deliver a Layer 3 provider-provisioned VPN service. This document is limited to BGP PE-based VPNs as described in RFCs 4026, 4110, and 4364. 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 the Layer 3 IP VPN service configuration components. It will be up to the management system to take this model as input and use specific configuration models to configure the different network elements to deliver the service. How the configuration of network elements is done is out of scope for this document.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8049.

Copyright Notice

Table of Contents

1. Introduction

This document defines a Layer 3 VPN service data model written in YANG. The model defines service configuration elements that can be used in communication protocols between customers and network operators. Those elements can also be used as input to automated control and configuration applications.

1.1. Terminology

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

- client
- configuration data
- server
- 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].

This document presents some configuration examples using XML representation.

1.2. Requirements Language

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

1.3. Tree Diagrams

A simplified graphical representation of the data model is presented in Section 6.

The meanings of the symbols in these diagrams are as follows:

- Brackets "[" and "]" enclose list keys.
- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Symbols after data node names: "?" means an optional node, and "*" denotes a "list" or "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Acronyms

AAA: Authentication, Authorization, and Accounting.

ACL: Access Control List.

ADSL: Asymmetric DSL.

AH: Authentication Header.

AS: Autonomous System.

ASBR: Autonomous System Border Router.

ASM: Any-Source Multicast.

BAS: Broadband Access Switch.

BFD: Bidirectional Forwarding Detection.

BGP: Border Gateway Protocol.

BSR: Bootstrap Router.

CE: Customer Edge.

CLI: Command Line Interface.

CsC: Carriers' Carriers.

CSP: Cloud Service Provider.

DHCP: Dynamic Host Configuration Protocol.

DSLAM: Digital Subscriber Line Access Multiplexer.

ESP: Encapsulating Security Payload.

GRE: Generic Routing Encapsulation.

IGMP: Internet Group Management Protocol.

LAN: Local Area Network.

MLD: Multicast Listener Discovery.

MTU: Maximum Transmission Unit.

NAT: Network Address Translation.

NETCONF: Network Configuration Protocol.

NNI: Network-to-Network Interface.

OAM: Operations, Administration, and Maintenance.

OSPF: Open Shortest Path First.

OSS: Operations Support System.

PE: Provider Edge.

PIM: Protocol Independent Multicast.

POP: Point of Presence.

QoS: Quality of Service.

RD: Route Distinguisher.

RIP: Routing Information Protocol.

RP: Rendezvous Point.

RT: Route Target.

SFTP: Secure FTP.

SLA: Service Level Agreement.

SLAAC: Stateless Address Autoconfiguration.

SP: Service Provider.

SPT: Shortest Path Tree.

SSM: Source-Specific Multicast.

VM: Virtual Machine.

VPN: Virtual Private Network.

VRF: VPN Routing and Forwarding.

VRRP: Virtual Router Redundancy Protocol.

3. Definitions

Customer Edge (CE) Device: A CE is equipment dedicated to a particular customer; it is directly connected (at Layer 3) 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.

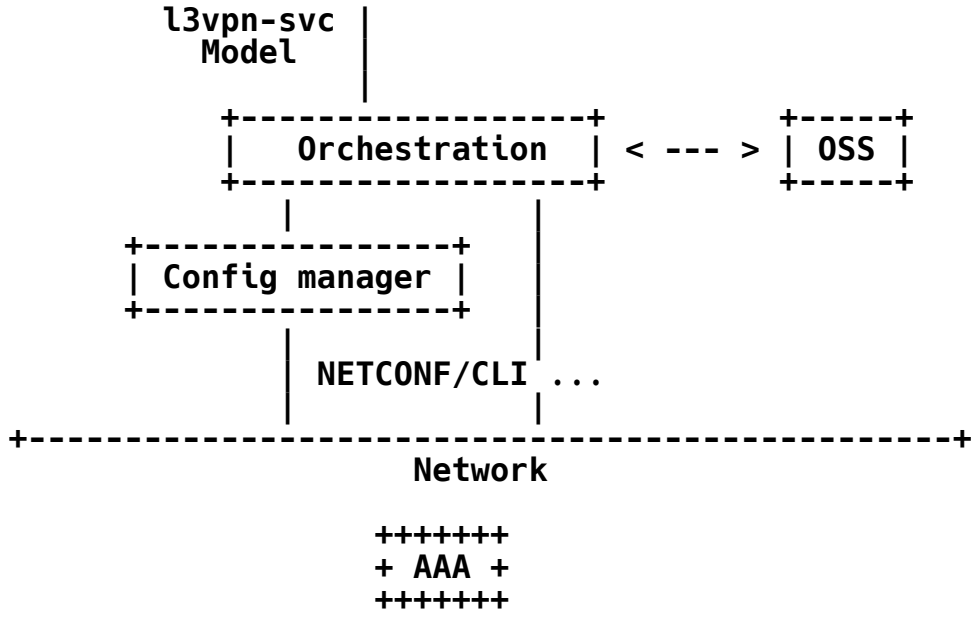
Provider Edge (PE) Device: A PE is equipment managed by the SP; it can support multiple VPNs for different customers and is directly connected (at Layer 3) to one or more CE devices via attachment circuits. A PE is usually located at an SP point of presence (POP) and is managed by the SP.

PE-Based VPNs: The PE devices know that certain traffic is VPN traffic. They forward the traffic (through tunnels) based on the destination IP address of the packet and, optionally, based on other information in the IP header of the packet. The PE devices are themselves the tunnel endpoints. The tunnels may make use of various encapsulations to send traffic over the SP network (such as, but not restricted to, GRE, IP-in-IP, IPsec, or MPLS tunnels).

4. Layer 3 IP VPN Service Model

A Layer 3 IP VPN service is a collection of sites that are authorized to exchange traffic between each other over a shared IP infrastructure. This Layer 3 VPN service model aims at providing a common understanding of how the corresponding IP VPN service is to be deployed over the shared infrastructure. This service model is limited to BGP PE-based VPNs as described in [RFC4026], [RFC4110], and [RFC4364].

5. Service Data Model Usage



The idea of the L3 IP VPN service model is to propose an abstracted interface between customers and network operators to manage configuration of components of an L3VPN service. A typical scenario would be to use this model as an input for an orchestration layer that will be responsible for translating it to an orchestrated configuration of network elements that will be part of the service. The network elements can be routers but can also be servers (like AAA); the network's configuration is not limited to these examples. 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.

The usage of this service model is not limited to this example; it can be used by any component of the management system but not directly by network elements.

6. Design of the Data Model

The YANG module is divided into two main containers: "vpn-services" and "sites".

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

A "site" is composed of at least one "site-network-access" and, in the case of multihoming, may have multiple site-network-access points. The site-network-access attachment is done through a "bearer" with an "ip-connection" on top. The bearer refers to properties of the attachment that are below Layer 3, while the connection refers to properties oriented to the Layer 3 protocol. The bearer may be allocated dynamically by the SP, and the customer may provide some constraints or parameters to drive the placement of the access.

Authorization of traffic exchange is done through what we call a VPN policy or VPN service topology defining routing exchange rules between sites.

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

```
module: ietf-l3vpn-svc
  +--rw l3vpn-svc
   +--rw vpn-services
   | +--rw vpn-service* [vpn-id]
   |   +--rw vpn-id         svc-id
   |   +--rw customer-name?     string
   |   +--rw vpn-service-topology?  identityref
   |   +--rw cloud-accesses {cloud-access}?
   |   | +--rw cloud-access* [cloud-identifier]
   |   |   +--rw cloud-identifier    string
   |   |   +--rw (list-flavor)?
   |   |   | +--:(permit-any)
   |   |   | | +--rw permit-any?      empty
   |   |   | +--:(deny-any-except)
   |   |   | | +--rw permit-site*      leafref
   |   |   | +--:(permit-any-except)
   |   |   |   +--rw deny-site*       leafref
   |   |   +--rw authorized-sites
   |   |   | +--rw authorized-site* [site-id]
   |   |   |   +--rw site-id  leafref
   |   |   +--rw denied-sites
   |   |   | +--rw denied-site* [site-id]
   |   |   |   +--rw site-id  leafref
   |   |   +--rw address-translation
```

```
|    |    +--rw nat44
|    |      +--rw enabled?         boolean
|    |      +--rw nat44-customer-address?  inet:ipv4-address
|    +--rw multicast {multicast}?
|    | +--rw enabled?         boolean
|    | +--rw customer-tree-flavors
|    | | +--rw tree-flavor*  identityref
|    | +--rw rp
|    |   +--rw rp-group-mappings
|    |   | +--rw rp-group-mapping* [id]
|    |   |   +--rw id         uint16
|    |   |   +--rw provider-managed
|    |   |   | +--rw enabled?          boolean
|    |   |   | +--rw rp-redundancy?       boolean
|    |   |   | +--rw optimal-traffic-delivery?  boolean
|    |   |   +--rw rp-address?     inet:ip-address
|    |   |   +--rw groups
|    |   |    +--rw group* [id]
|    |   |      +--rw id        uint16
|    |   |      +--rw (group-format)?
|    |   |       +--:(startend)
|    |   |       | +--rw group-start?   inet:ip-address
|    |   |       | +--rw group-end?    inet:ip-address
|    |   |       +--:(singleaddress)
|    |   |         +--rw group-address?  inet:ip-address
|    |   +--rw rp-discovery
|    |    +--rw rp-discovery-type?  identityref
|    |    +--rw bsr-candidates
|    |      +--rw bsr-candidate-address*  inet:ip-address
|    +--rw carrierscarrier?    boolean {carrierscarrier}?
|    +--rw extranet-vpns {extranet-vpn}?
|     +--rw extranet-vpn* [vpn-id]
|       +--rw vpn-id       svc-id
|       +--rw local-sites-role?  identityref
+--rw sites
  +--rw site* [site-id]
   +--rw site-id         svc-id
   +--rw requested-site-start?  yang:date-and-time
   +--rw requested-site-stop?   yang:date-and-time
   +--rw locations
   | +--rw location* [location-id]
   |   +--rw location-id   svc-id
   |   +--rw address?    string
   |   +--rw postal-code?  string
   |   +--rw state?     string
   |   +--rw city?      string
   |   +--rw country-code?  string
```

```
        +--rw devices
        | +--rw device* [device-id]
        |   +--rw device-id   svc-id
        |   +--rw location?   leafref
        |   +--rw management
        |    +--rw address-family?  address-family
        |    +--rw address?     inet:ip-address
        +--rw site-diversity {site-diversity}?
        | +--rw groups
        |   +--rw group* [group-id]
        |    +--rw group-id  string
        +--rw management
        | +--rw type?  identityref
        +--rw vpn-policies
        | +--rw vpn-policy* [vpn-policy-id]
        |   +--rw vpn-policy-id  svc-id
        |   +--rw entries* [id]
        |    +--rw id    svc-id
        |    +--rw filter
        |    | +--rw (lan)?
        |    |   +--:(prefixes)
        |    |   | +--rw ipv4-lan-prefix*  inet:ipv4-prefix {ipv4}?
        |    |   | +--rw ipv6-lan-prefix*  inet:ipv6-prefix {ipv6}?
        |    |   +--:(lan-tag)
        |    |    +--rw lan-tag*      string
        |    +--rw vpn
        |      +--rw vpn-id    leafref
        |      +--rw site-role?  identityref
        +--rw site-vpn-flavor?     identityref
        +--rw maximum-routes
        | +--rw address-family* [af]
        |   +--rw af        address-family
        |   +--rw maximum-routes?  uint32
        +--rw security
        | +--rw authentication
        | +--rw encryption {encryption}?
        |   +--rw enabled?       boolean
        |   +--rw layer         enumeration
        |   +--rw encryption-profile
        |    +--rw (profile)?
        |      +--:(provider-profile)
        |      | +--rw profile-name?  string
        |      +--:(customer-profile)
        |       +--rw algorithm?    string
        |       +--rw (key-type)?
        |         +--:(psk)
        |         | +--rw preshared-key?  string
        |         +--:(pki)
```

```
+--rw service
| +--rw qos {qos}?
| | +--rw qos-classification-policy
| | | +--rw rule* [id]
| | |   +--rw id          uint16
| | |   +--rw (match-type)?
| | |    | +--:(match-flow)
| | |    | | +--rw match-flow
| | |    | |   +--rw dscp?        inet:dscp
| | |    | |   +--rw dot1p?        uint8
| | |    | |   +--rw ipv4-src-prefix?   inet:ipv4-prefix
| | |    | |   +--rw ipv6-src-prefix?   inet:ipv6-prefix
| | |    | |   +--rw ipv4-dst-prefix?   inet:ipv4-prefix
| | |    | |   +--rw ipv6-dst-prefix?   inet:ipv6-prefix
| | |    | |   +--rw l4-src-port?     inet:port-number
| | |    | |   +--rw target-sites*    svc-id
| | |    | |   +--rw l4-src-port-range
| | |    | |   | +--rw lower-port?  inet:port-number
| | |    | |   | +--rw upper-port?  inet:port-number
| | |    | |   +--rw l4-dst-port?     inet:port-number
| | |    | |   +--rw l4-dst-port-range
| | |    | |   | +--rw lower-port?  inet:port-number
| | |    | |   | +--rw upper-port?  inet:port-number
| | |    | |   +--rw protocol-field?   union
| | |    | +--:(match-application)
| | |    |   +--rw match-application?  identityref
| | |   +--rw target-class-id?   string
| |  +--rw qos-profile
| |    +--rw (qos-profile)?
| |     +--:(standard)
| |     | +--rw profile?  string
| |     +--:(custom)
| |       +--rw classes {qos-custom}?
| |        +--rw class* [class-id]
| |          +--rw class-id   string
| |          +--rw rate-limit?  uint8
| |          +--rw latency
| |          | +--rw (flavor)?
| |          |    ...
| |          +--rw jitter
| |          | +--rw (flavor)?
| |          |    ...
| |          +--rw bandwidth
| |           +--rw guaranteed-bw-percent?  uint8
| |           +--rw end-to-end?       empty
| +--rw carrierscarrier {carrierscarrier}?
| | +--rw signalling-type?  enumeration
```

```
|  +--rw multicast {multicast}?
|     +--rw multicast-site-type?    enumeration
|     +--rw multicast-address-family
|     |  +--rw ipv4?  boolean {ipv4}?
|     |  +--rw ipv6?  boolean {ipv6}?
|     +--rw protocol-type?       enumeration
+--rw traffic-protection {fast-reroute}?
|  +--rw enabled?  boolean
+--rw routing-protocols
|  +--rw routing-protocol* [type]
|     +--rw type   identityref
|     +--rw ospf {rtg-ospf}?
|     |  +--rw address-family*  address-family
|     |  +--rw area-address?   yang:dotted-quad
|     |  +--rw metric?      uint16
|     |  +--rw sham-links {rtg-ospf-sham-link}?
|     |     +--rw sham-link* [target-site]
|     |        +--rw target-site  svc-id
|     |        +--rw metric?    uint16
|     +--rw bgp {rtg-bgp}?
|     |  +--rw autonomous-system?  uint32
|     |  +--rw address-family*   address-family
|     +--rw static
|     |  +--rw cascaded-lan-prefixes
|     |     +--rw ipv4-lan-prefixes* [lan next-hop] {ipv4}?
|     |     |  +--rw lan     inet:ipv4-prefix
|     |     |  +--rw lan-tag?  string
|     |     |  +--rw next-hop  inet:ipv4-address
|     |     +--rw ipv6-lan-prefixes* [lan next-hop] {ipv6}?
|     |        +--rw lan     inet:ipv6-prefix
|     |        +--rw lan-tag?  string
|     |        +--rw next-hop  inet:ipv6-address
|     +--rw rip {rtg-rip}?
|     |  +--rw address-family*  address-family
|     +--rw vrrp {rtg-vrrp}?
|        +--rw address-family*  address-family
+--ro actual-site-start?    yang:date-and-time
+--ro actual-site-stop?    yang:date-and-time
+--rw site-network-accesses
   +--rw site-network-access* [site-network-access-id]
      +--rw site-network-access-id   svc-id
      +--rw site-network-access-type?  identityref
      +--rw (location-flavor)
      |  +--:(location)
      |  |  +--rw location-reference?     leafref
      |  +--:(device)
      |     +--rw device-reference?      leafref
```

```
          +--rw access-diversity {site-diversity}?
          | +--rw groups
          | | +--rw group* [group-id]
          | |   +--rw group-id  string
          | +--rw constraints
          |   +--rw constraint* [constraint-type]
          |    +--rw constraint-type  identityref
          |    +--rw target
          |      +--rw (target-flavor)?
          |       +--:(id)
          |       | +--rw group* [group-id]
          |       |    ...
          |       +--:(all-accesses)
          |       | +--rw all-other-accesses?  empty
          |       +--:(all-groups)
          |         +--rw all-other-groups?   empty
          +--rw bearer
          | +--rw requested-type {requested-type}?
          | | +--rw requested-type?  string
          | | +--rw strict?      boolean
          | +--rw always-on?     boolean {always-on}?
          | +--rw bearer-reference?  string {bearer-reference}?
          +--rw ip-connection
          | +--rw ipv4 {ipv4}?
          | | +--rw address-allocation-type?   identityref
          | | +--rw number-of-dynamic-address?  uint8
          | | +--rw dhcp-relay
          | | | +--rw customer-dhcp-servers
          | | |   +--rw server-ip-address*  inet:ipv4-address
          | | +--rw addresses
          | |   +--rw provider-address?  inet:ipv4-address
          | |   +--rw customer-address?  inet:ipv4-address
          | |   +--rw mask?        uint8
          | +--rw ipv6 {ipv6}?
          | | +--rw address-allocation-type?   identityref
          | | +--rw number-of-dynamic-address?  uint8
          | | +--rw dhcp-relay
          | | | +--rw customer-dhcp-servers
          | | |   +--rw server-ip-address*  inet:ipv6-address
          | | +--rw addresses
          | |   +--rw provider-address?  inet:ipv6-address
          | |   +--rw customer-address?  inet:ipv6-address
          | |   +--rw mask?        uint8
```

```
|  +--rw oam
|    +--rw bfd {bfd}?
|     +--rw enabled?    boolean
|     +--rw (holdtime)?
|       +--:(profile)
|       | +--rw profile-name?  string
|       +--:(fixed)
|         +--rw fixed-value?  uint32
+--rw security
|  +--rw authentication
|  +--rw encryption {encryption}?
|    +--rw enabled?       boolean
|    +--rw layer          enumeration
|    +--rw encryption-profile
|     +--rw (profile)?
|       +--:(provider-profile)
|       | +--rw profile-name?  string
|       +--:(customer-profile)
|         +--rw algorithm?    string
|         +--rw (key-type)?
|           +--:(psk)
|           |   ...
|           +--:(pki)
+--rw service
|  +--rw svc-input-bandwidth?  uint32
|  +--rw svc-output-bandwidth?  uint32
|  +--rw svc-mtu?         uint16
|  +--rw qos {qos}?
|  | +--rw qos-classification-policy
|  | | +--rw rule* [id]
|  | |   +--rw id          uint16
|  | |   +--rw (match-type)?
|  | |   | +--:(match-flow)
|  | |   | | +--rw match-flow
|  | |   | |    ...
|  | |   | +--:(match-application)
|  | |   |   +--rw match-application?  identityref
|  | |   +--rw target-class-id?   string
|  | +--rw qos-profile
|  |   +--rw (qos-profile)?
|  |    +--:(standard)
|  |    | +--rw profile?  string
|  |    +--:(custom)
|  |      +--rw classes {qos-custom}?
|  |       +--rw class* [class-id]
|  |          ...
```

```
|  +--rw carrierscarrier {carrierscarrier}?
|  |  +--rw signalling-type?  enumeration
|  +--rw multicast {multicast}?
|     +--rw multicast-site-type?    enumeration
|     +--rw multicast-address-family
|     |  +--rw ipv4?  boolean {ipv4}?
|     |  +--rw ipv6?  boolean {ipv6}?
|     +--rw protocol-type?       enumeration
+--rw routing-protocols
|  +--rw routing-protocol* [type]
|     +--rw type   identityref
|     +--rw ospf {rtg-ospf}?
|     |  +--rw address-family*  address-family
|     |  +--rw area-address?   yang:dotted-quad
|     |  +--rw metric?      uint16
|     |  +--rw sham-links {rtg-ospf-sham-link}?
|     |     +--rw sham-link* [target-site]
|     |        +--rw target-site  svc-id
|     |        +--rw metric?    uint16
|     +--rw bgp {rtg-bgp}?
|     |  +--rw autonomous-system?  uint32
|     |  +--rw address-family*   address-family
|     +--rw static
|     |  +--rw cascaded-lan-prefixes
|     |     +--rw ipv4-lan-prefixes* [lan next-hop] {ipv4}?
|     |     |  +--rw lan     inet:ipv4-prefix
|     |     |  +--rw lan-tag?  string
|     |     |  +--rw next-hop  inet:ipv4-address
|     |     +--rw ipv6-lan-prefixes* [lan next-hop] {ipv6}?
|     |        +--rw lan     inet:ipv6-prefix
|     |        +--rw lan-tag?  string
|     |        +--rw next-hop  inet:ipv6-address
|     +--rw rip {rtg-rip}?
|     |  +--rw address-family*  address-family
|     +--rw vrrp {rtg-vrrp}?
|        +--rw address-family*  address-family
+--rw availability
|  +--rw access-priority?  uint32
+--rw vpn-attachment
   +--rw (attachment-flavor)
      +--:(vpn-policy-id)
      |  +--rw vpn-policy-id?  leafref
      +--:(vpn-id)
         +--rw vpn-id?     leafref
         +--rw site-role?    identityref
```

6.1. Features and Augmentation

The model defined in this document implements many features that allow implementations to be modular. As an example, an implementation may support only IPv4 VPNs (IPv4 feature), IPv6 VPNs (IPv6 feature), or both (by advertising both features). The routing protocols proposed to the customer may also be enabled through features. This model also proposes some features for options that are more advanced, such as support for extranet VPNs (Section 6.2.4), site diversity (Section 6.6), and QoS (Section 6.12.2).

In addition, as for any YANG model, this service model can be augmented to implement new behaviors or specific features. For example, this model proposes different options for IP address assignments; if those options do not fulfill all requirements, new options can be added through augmentation.

6.2. VPN Service Overview

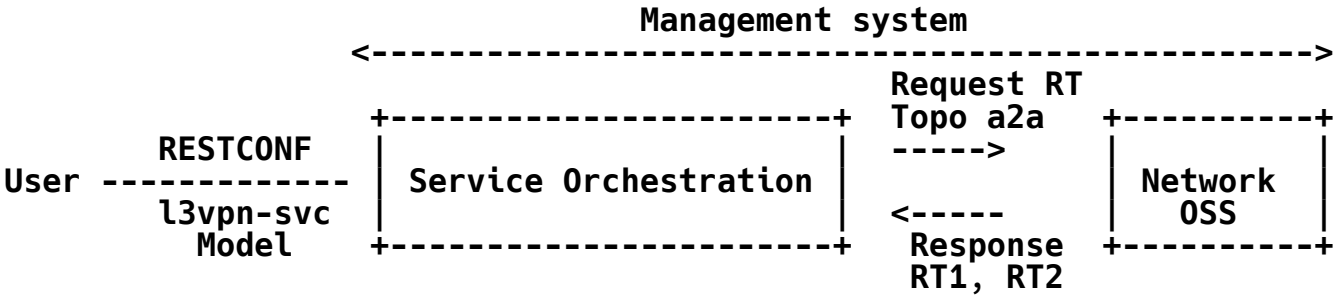
A vpn-service list item contains generic information about the VPN service. The "vpn-id" provided in the vpn-service list refers to an internal reference for this VPN service, while the customer name refers to a more-explicit reference to the customer. This identifier is purely internal to the organization responsible for the VPN service.

6.2.1. VPN Service Topology

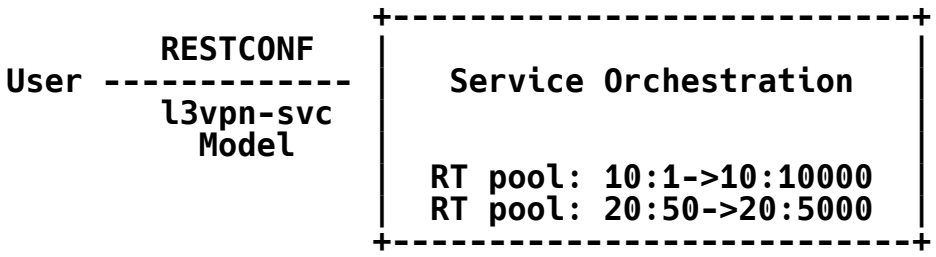
The type of VPN service topology is required for configuration. Our proposed model supports any-to-any, Hub and Spoke (where Hubs can exchange traffic), and "Hub and Spoke disjoint" (where Hubs cannot exchange traffic). New topologies could be added via augmentation. By default, the any-to-any VPN service topology is used.

6.2.1.1. Route Target Allocation

A Layer 3 PE-based VPN is built using route targets (RTs) as described in [RFC4364]. The management system is expected to automatically allocate a set of RTs upon receiving a VPN service creation request. How the management system allocates RTs is out of scope for this document, but multiple ways could be envisaged, as described below.



In the example above, a service orchestration, owning the instantiation of this service model, requests RTs to the network OSS. Based on the requested VPN service topology, the network OSS replies with one or multiple RTs. The interface between this service orchestration and the network OSS is out of scope for this document.



In the example above, a service orchestration, owning the instantiation of this service model, owns one or more pools of RTs (specified by the SP) that can be allocated. Based on the requested VPN service topology, it will allocate one or multiple RTs from the pool.

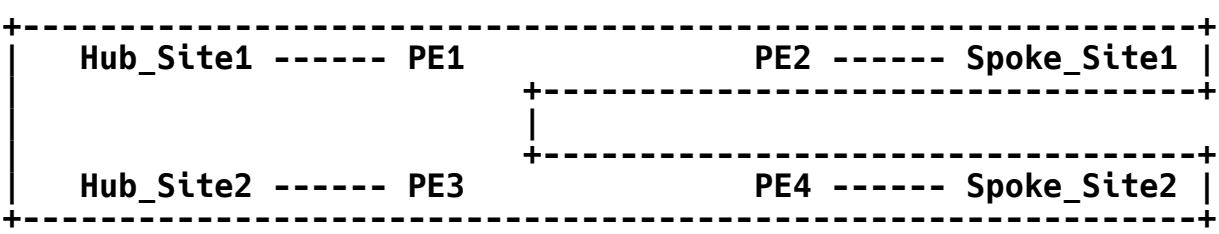
The mechanisms shown above are just examples and should not be considered an exhaustive list of solutions.

6.2.1.2. Any-to-Any

Any-to-Any VPN Service Topology

In the any-to-any VPN service topology, all VPN sites can communicate with each other without any restrictions. The management system that receives an any-to-any IP VPN service request through this model is expected to assign and then configure the VRF and RTs on the appropriate PEs. In the any-to-any case, a single RT is generally required, and every VRF imports and exports this RT.

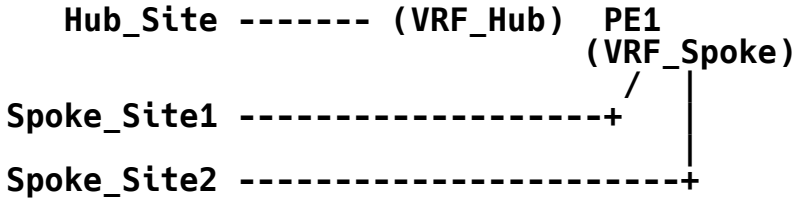
6.2.1.3. Hub and Spoke



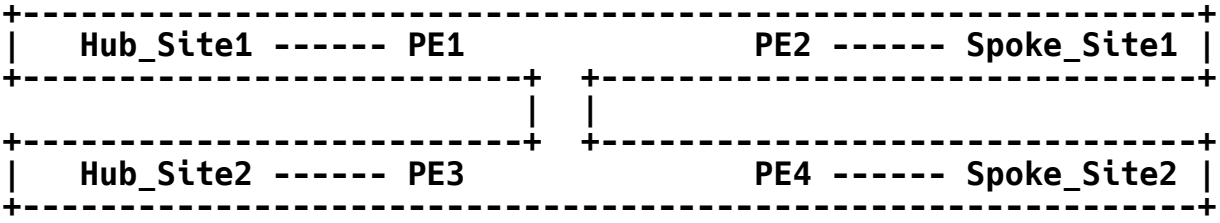
Hub-and-Spoke VPN Service Topology

In the Hub-and-Spoke VPN service topology, all Spoke sites can communicate only with Hub sites but not with each other, and Hubs can also communicate with each other. The management system that owns an any-to-any IP VPN service request through this model is expected to assign and then configure the VRF and RTs on the appropriate PEs. In the Hub-and-Spoke case, two RTs are generally required (one RT for Hub routes and one RT for Spoke routes). A Hub VRF that connects Hub sites will export Hub routes with the Hub RT and will import Spoke routes through the Spoke RT. It will also import the Hub RT to allow Hub-to-Hub communication. A Spoke VRF that connects Spoke sites will export Spoke routes with the Spoke RT and will import Hub routes through the Hub RT.

The management system MUST take into account constraints on Hub-and-Spoke connections. For example, if a management system decides to mesh a Spoke site and a Hub site on the same PE, it needs to mesh connections in different VRFs, as shown in the figure below.



6.2.1.4. Hub and Spoke Disjoint



Hub and Spoke Disjoint VPN Service Topology

In the Hub and Spoke disjoint VPN service topology, all Spoke sites can communicate only with Hub sites but not with each other, and Hubs cannot communicate with each other. The management system that owns an any-to-any IP VPN service request through this model is expected to assign and then configure the VRF and RTs on the appropriate PEs. In the Hub-and-Spoke case, two RTs are required (one RT for Hub routes and one RT for Spoke routes). A Hub VRF that connects Hub sites will export Hub routes with the Hub RT and will import Spoke routes through the Spoke RT. A Spoke VRF that connects Spoke sites will export Spoke routes with the Spoke RT and will import Hub routes through the Hub RT.

The management system MUST take into account constraints on Hub-and-Spoke connections, as in the previous case.

Hub and Spoke disjoint can also be seen as multiple Hub-and-Spoke VPNs (one per Hub) that share a common set of Spoke sites.

6.2.2. Cloud Access

The proposed model provides cloud access configuration via the "cloud-accesses" container. The usage of cloud-access is targeted for the public cloud. An Internet access can also be considered a public cloud access service. The cloud-accesses container provides parameters for network address translation and authorization rules.

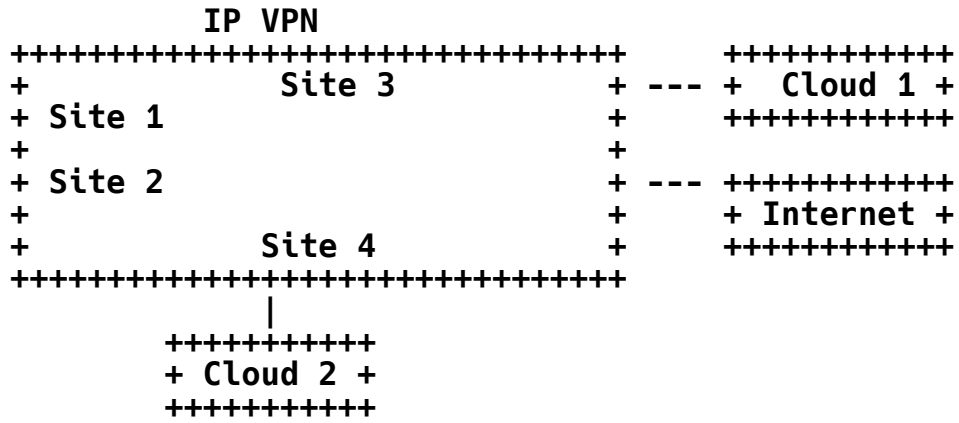
A private cloud access may be addressed through NNIs, as described in Section 6.15.

A cloud identifier is used to reference the target service. This identifier is local to each administration.

The model allows for source address translation before accessing the cloud. IPv4-to-IPv4 address translation (NAT44) is the only supported option, but other options can be added through augmentation. If IP source address translation is required to access the cloud, the "enabled" leaf MUST be set to true in the "nat44" container. An IP address may be provided in the "customer-address" leaf if the customer is providing the IP address to be used for the cloud access. If the SP is providing this address, "customer-address" is not necessary, as it can be picked from a pool of SPs.

By default, all sites in the IP VPN MUST be authorized to access the cloud. If restrictions are required, a user MAY configure the "permit-site" or "deny-site" leaf-list. The permit-site leaf-list defines the list of sites authorized for cloud access. The deny-site leaf-list defines the list of sites denied for cloud access. The model supports both "deny-any-except" and "permit-any-except" authorization.

How the restrictions will be configured on network elements is out of scope for this document.



In the example above, we configure the global VPN to access the Internet by creating a cloud-access pointing to the cloud identifier for the Internet service. No authorized sites will be configured, as all sites are required to access the Internet. The "address-translation/nat44/enabled" leaf will be set to true.

```
<vpn-service>
    <vpn-id>123456487</vpn-id>
    <cloud-accesses>
     <cloud-access>
        <cloud-identifier>INTERNET</cloud-identifier>
        <address-translation>
          <nat44>
            <enabled>true</enabled>
          </nat44>
        </address-translation>
     </cloud-access>
    </cloud-accesses>
</vpn-service>
```

If Site 1 and Site 2 require access to Cloud 1, a new cloud-access pointing to the cloud identifier of Cloud 1 will be created. The permit-site leaf-list will be filled with a reference to Site 1 and Site 2.

```
<vpn-service>
    <vpn-id>123456487</vpn-id>
    <cloud-accesses>
     <cloud-access>
        <cloud-identifier>Cloud1</cloud-identifier>
        <permit-site>site1</permit-site>
        <permit-site>site2</permit-site>
     </cloud-access>
    </cloud-accesses>
</vpn-service>
```

If all sites except Site 1 require access to Cloud 2, a new cloud-access pointing to the cloud identifier of Cloud 2 will be created. The deny-site leaf-list will be filled with a reference to Site 1.

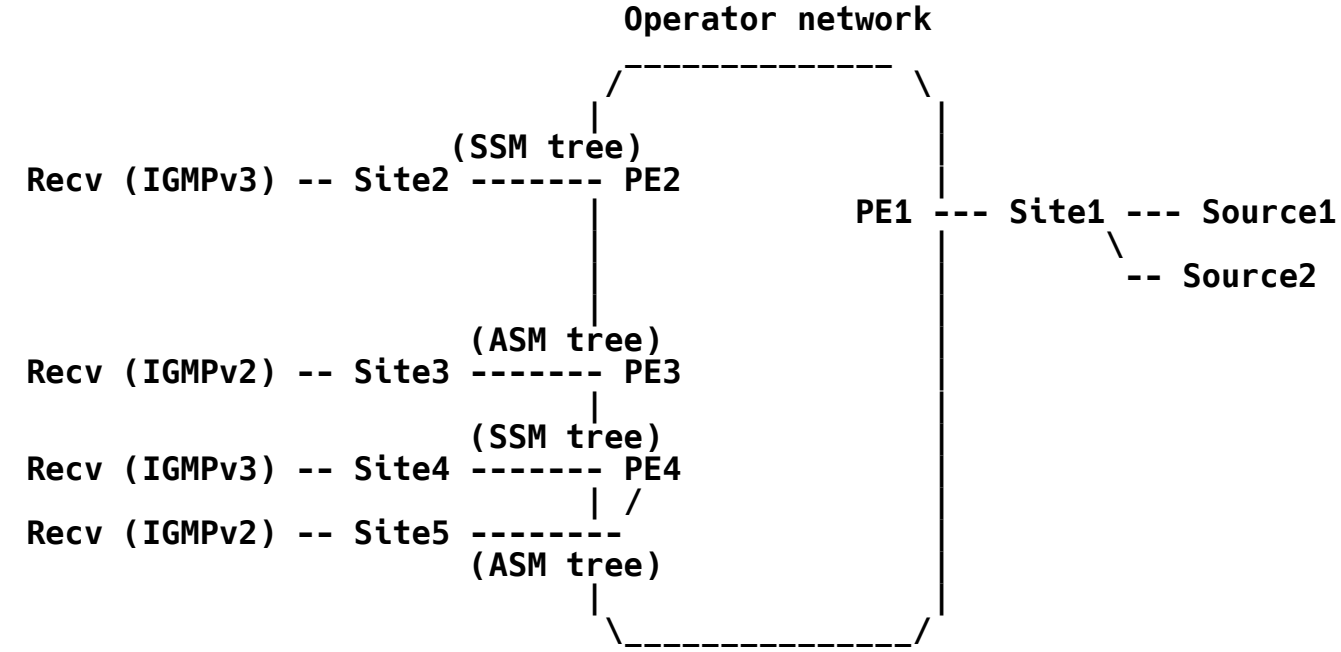
```
<vpn-service>
    <vpn-id>123456487</vpn-id>
    <cloud-accesses>
     <cloud-access>
        <cloud-identifier>Cloud2</cloud-identifier>
        <deny-site>site1</deny-site>
     </cloud-access>
    </cloud-accesses>
</vpn-service>
```

6.2.3. Multicast Service

Multicast in IP VPNs is described in [RFC6513].

If multicast support is required for an IP VPN, some global multicast parameters are required as input for the service request.

Users of this model will need to provide the flavors of trees that will be used by customers within the IP VPN (customer tree). The proposed model supports bidirectional, shared, and source-based trees (and can be augmented). Multiple flavors of trees can be supported simultaneously.



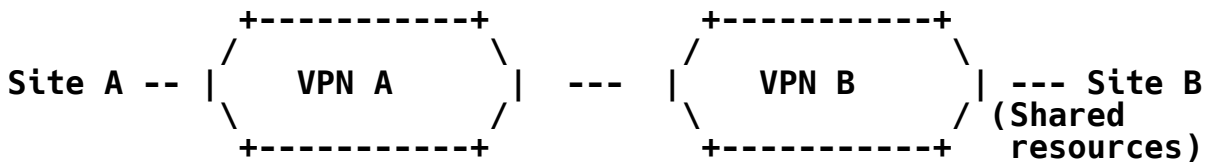
When an ASM flavor is requested, this model requires that the "rp" and "rp-discovery" parameters be filled. Multiple RP-to-group mappings can be created using the "rp-group-mappings" container. For each mapping, the SP can manage the RP service by setting the "provider-managed/enabled" leaf to true. In the case of a provider-managed RP, the user can request RP redundancy and/or optimal traffic delivery. Those parameters will help the SP select the appropriate technology or architecture to fulfill the customer service requirement: for instance, in the case of a request for optimal traffic delivery, an SP may use Anycast-RP or RP-tree-to-SPT switchover architectures.

In the case of a customer-managed RP, the RP address must be filled in the RP-to-group mappings using the "rp-address" leaf. This leaf is not needed for a provider-managed RP.

Users can define a specific mechanism for RP discovery, such as the "auto-rp", "static-rp", or "bsr-rp" modes. By default, the model uses "static-rp" if ASM is requested. A single rp-discovery mechanism is allowed for the VPN. The "rp-discovery" container can be used for both provider-managed and customer-managed RPs. In the case of a provider-managed RP, if the user wants to use "bsr-rp" as a discovery protocol, an SP should consider the provider-managed "rp-group-mappings" for the "bsr-rp" configuration. The SP will then configure its selected RPs to be "bsr-rp-candidates". In the case of a customer-managed RP and a "bsr-rp" discovery mechanism, the "rp-address" provided will be the bsr-rp candidate.

6.2.4. Extranet VPNs

There are some cases where a particular VPN needs access to resources (servers, hosts, etc.) that are external. Those resources may be located in another VPN.



In the figure above, VPN B has some resources on Site B that need to be available to some customers/partners. VPN A must be able to access those VPN B resources.

Such a VPN connection scenario can be achieved via a VPN policy as defined in Section 6.5.2.2. But there are some simple cases where a particular VPN (VPN A) needs access to all resources in another VPN (VPN B). The model provides an easy way to set up this connection using the "extranet-vpns" container.

The extranet-vpns container defines a list of VPNs a particular VPN wants to access. The extranet-vpns container must be used on customer VPNs accessing extranet resources in another VPN. In the figure above, in order to provide VPN A with access to VPN B, the extranet-vpns container needs to be configured under VPN A with an entry corresponding to VPN B. There is no service configuration requirement on VPN B.

Readers should note that even if there is no configuration requirement on VPN B, if VPN A lists VPN B as an extranet, all sites in VPN B will gain access to all sites in VPN A.

The "site-role" leaf defines the role of the local VPN sites in the target extranet VPN service topology. Site roles are defined in Section 6.4. Based on this, the requirements described in Section 6.4 regarding the site-role leaf are also applicable here.

In the example below, VPN A accesses VPN B resources through an extranet connection. A Spoke role is required for VPN A sites, as sites from VPN A must not be able to communicate with each other through the extranet VPN connection.

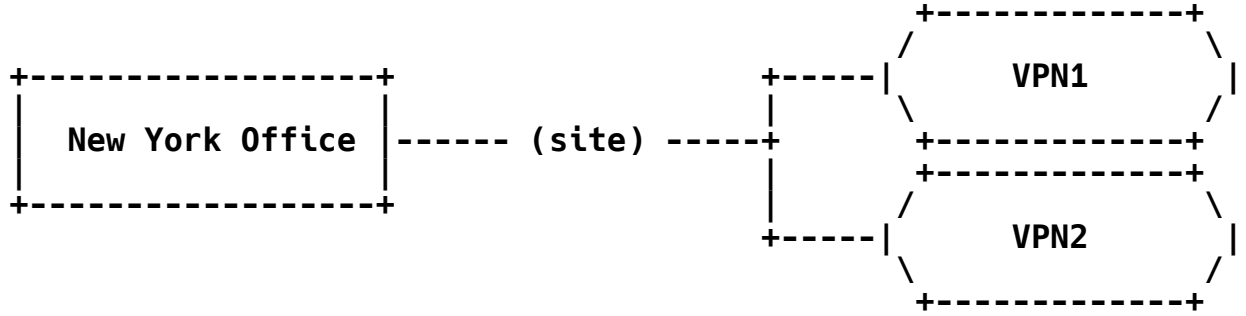
```
<vpn-service>
    <vpn-id>VPNB</vpn-id>
    <vpn-service-topology>hub-spoke</vpn-service-topology>
</vpn-service>
<vpn-service>
    <vpn-id>VPNA</vpn-id>
    <vpn-service-topology>any-to-any</vpn-service-topology>
    <extranet-vpns>
        <extranet-vpn>
            <vpn-id>VPNB</vpn-id>
            <site-role>spoke-role</site-role>
        </extranet-vpn>
    </extranet-vpns>
</vpn-service>
```

This model does not define how the extranet configuration will be achieved.

Any VPN interconnection scenario that is more complex (e.g., only certain parts of sites on VPN A accessing only certain parts of sites on VPN B) needs to be achieved using a VPN attachment as defined in Section 6.5.2, and especially a VPN policy as defined in Section 6.5.2.2.

6.3. Site Overview

A site represents a connection of a customer office to one or more VPN services.



A site has several characteristics:

- Unique identifier (site-id): uniquely identifies the site within the overall network infrastructure. The identifier is a string that allows any encoding for the local administration of the VPN service.

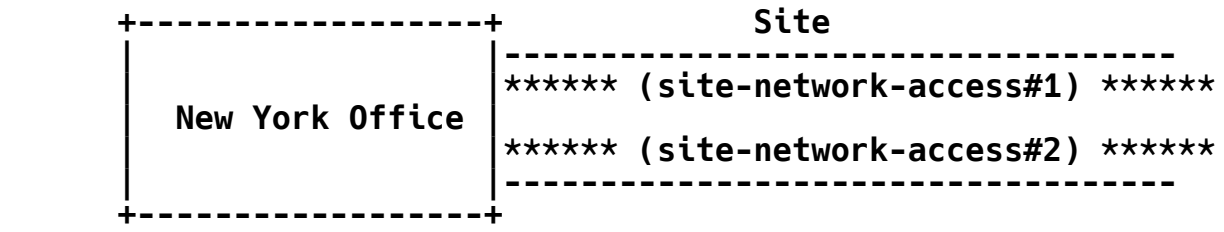
- Locations (locations): site location information that allows easy retrieval of information from the nearest available resources. A site may be composed of multiple locations.

- Devices (devices): allows the customer to request one or more customer premises equipment entities from the SP for a particular site.

- Management (management): defines the type of management for the site -- for example, co-managed, customer-managed, or provider-managed. See Section 6.10.

- Site network accesses (site-network-accesses): defines the list of network accesses associated with the sites, and their properties -- especially bearer, connection, and service parameters.

A site-network-access represents an IP logical connection of a site. A site may have multiple site-network-accesses.

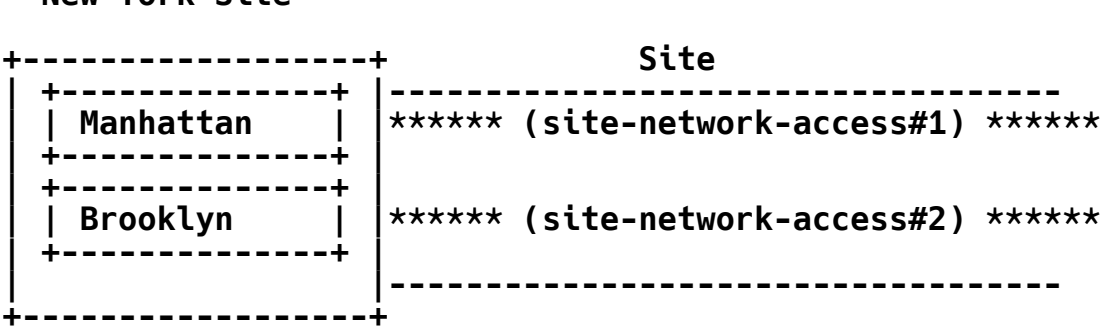


Multiple site-network-accesses are used, for instance, in the case of multihoming. Some other meshing cases may also include multiple site-network-accesses.

The site configuration is viewed as a global entity; we assume that it is mostly the management system's role to split the parameters between the different elements within the network. For example, in the case of the site-network-access configuration, the management system needs to split the overall parameters between the PE configuration and the CE configuration.

6.3.1. Devices and Locations

A site may be composed of multiple locations. All the locations will need to be configured as part of the "locations" container and list. A typical example of a multi-location site is a headquarters office in a city composed of multiple buildings. Those buildings may be located in different parts of the city and may be linked by intra-city fibers (customer metropolitan area network). In such a case, when connecting to a VPN service, the customer may ask for multihoming based on its distributed locations.



A customer may also request some premises equipment entities (CEs) from the SP via the "devices" container. Requesting a CE implies a provider-managed or co-managed model. A particular device must be ordered to a particular already-configured location. This would help the SP send the device to the appropriate postal address. In a multi-location site, a customer may, for example, request a CE for each location on the site where multihoming must be implemented. In the figure above, one device may be requested for the Manhattan location and one other for the Brooklyn location.

By using devices and locations, the user can influence the multihoming scenario he wants to implement: single CE, dual CE, etc.

6.3.2. Site Network Accesses

As mentioned earlier, a site may be multihomed. Each IP network access for a site is defined in the "site-network-accesses" container. The site-network-access parameter defines how the site is connected on the network and is split into three main classes of parameters:

- bearer: defines requirements of the attachment (below Layer 3).
- connection: defines Layer 3 protocol parameters of the attachment.
- availability: defines the site's availability policy. The availability parameters are defined in Section 6.7.

The site-network-access has a specific type (site-network-access-type). This document defines two types:

- point-to-point: describes a point-to-point connection between the SP and the customer.
- multipoint: describes a multipoint connection between the SP and the customer.

The type of site-network-access may have an impact on the parameters offered to the customer, e.g., an SP may not offer encryption for multipoint accesses. It is up to the provider to decide what parameter is supported for point-to-point and/or multipoint accesses; this topic is out of scope for this document. Some containers proposed in the model may require extensions in order to work properly for multipoint accesses.

6.3.2.1. Bearer

The bearer container defines the requirements for the site attachment to the provider network that are below Layer 3.

The bearer parameters will help determine the access media to be used. This is further described in Section 6.6.3.

6.3.2.2. Connection

The "ip-connection" container defines the protocol parameters of the attachment (IPv4 and IPv6). Depending on the management mode, it refers to PE-CE addressing or CE-to-customer-LAN addressing. In any case, it describes the responsibility boundary between the provider and the customer. For a customer-managed site, it refers to the PE-CE connection. For a provider-managed site, it refers to the CE-to-LAN connection.

6.3.2.2.1. IP Addressing

An IP subnet can be configured for either IPv4 or IPv6 Layer 3 protocols. For a dual-stack connection, two subnets will be provided, one for each address family.

The "address-allocation-type" determines how the address allocation needs to be done. The current model proposes five ways to perform IP address allocation:

- provider-dhcp: The provider will provide DHCP service for customer equipment; this is applicable to either the "IPv4" container or the "IPv6" container.
- provider-dhcp-relay: The provider will provide DHCP relay service for customer equipment; this is applicable to both IPv4 and IPv6 addressing. The customer needs to populate the DHCP server list to be used.
- static-address: Addresses will be assigned manually; this is applicable to both IPv4 and IPv6 addressing.
- slaac: This parameter enables stateless address autoconfiguration [RFC4862]. This is applicable to IPv6 only.
- provider-dhcp-slaac: The provider will provide DHCP service for customer equipment, as well as stateless address autoconfiguration. This is applicable to IPv6 only.

In the dynamic addressing mechanism, the SP is expected to provide at least the IP address, mask, and default gateway information.

6.3.2.2.2. OAM

A customer may require a specific IP connectivity fault detection mechanism on the IP connection. The model supports BFD as a fault detection mechanism. This can be extended with other mechanisms via augmentation. The provider can propose some profiles to the

customer, depending on the service level the customer wants to achieve. Profile names must be communicated to the customer. This communication is out of scope for this document. Some fixed values for the holdtime period may also be imposed by the customer if the provider allows the customer this function.

The "oam" container can easily be augmented by other mechanisms; in particular, work done by the LIME Working Group (https://datatracker.ietf.org/wg/lime/charter/) may be reused in applicable scenarios.

6.3.2.3. Inheritance of Parameters Defined at Site Level and Site Network Access Level

Some parameters can be configured at both the site level and the site-network-access level, e.g., routing, services, security. Inheritance applies when parameters are defined at the site level. If a parameter is configured at both the site level and the access level, the access-level parameter MUST override the site-level parameter. Those parameters will be described later in this document.

In terms of provisioning impact, it will be up to the implementation to decide on the appropriate behavior when modifying existing configurations. But the SP will need to communicate to the user about the impact of using inheritance. For example, if we consider that a site has already provisioned three site-network-accesses, what will happen if a customer changes a service parameter at the site level? An implementation of this model may update the service parameters of all already-provisioned site-network-accesses (with potential impact on live traffic), or it may take into account this new parameter only for the new sites.

6.4. Site Role

A VPN has a particular service topology, as described in Section 6.2.1. As a consequence, each site belonging to a VPN is assigned with a particular role in this topology. The site-role leaf defines the role of the site in a particular VPN topology.

In the any-to-any VPN service topology, all sites MUST have the same role, which will be "any-to-any-role".

In the Hub-and-Spoke VPN service topology or the Hub and Spoke disjoint VPN service topology, sites MUST have a Hub role or a Spoke role.

6.5. Site Belonging to Multiple VPNs

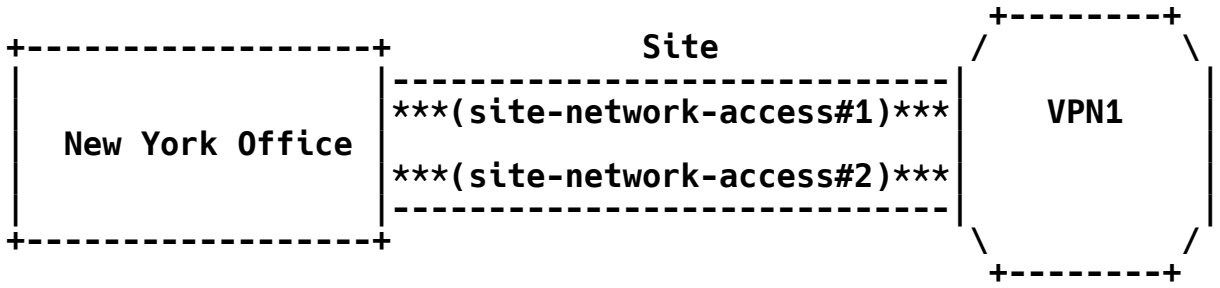
6.5.1. Site VPN Flavor

A site may be part of one or multiple VPNs. The "site-vpn-flavor" defines the way the VPN multiplexing is done. The current version of the model supports four flavors:

- site-vpn-flavor-single: The site belongs to only one VPN.
- site-vpn-flavor-multi: The site belongs to multiple VPNs, and all the logical accesses of the sites belong to the same set of VPNs.
- site-vpn-flavor-sub: The site belongs to multiple VPNs with multiple logical accesses. Each logical access may map to different VPNs (one or many).
- site-vpn-flavor-nni: The site represents an option A NNI.

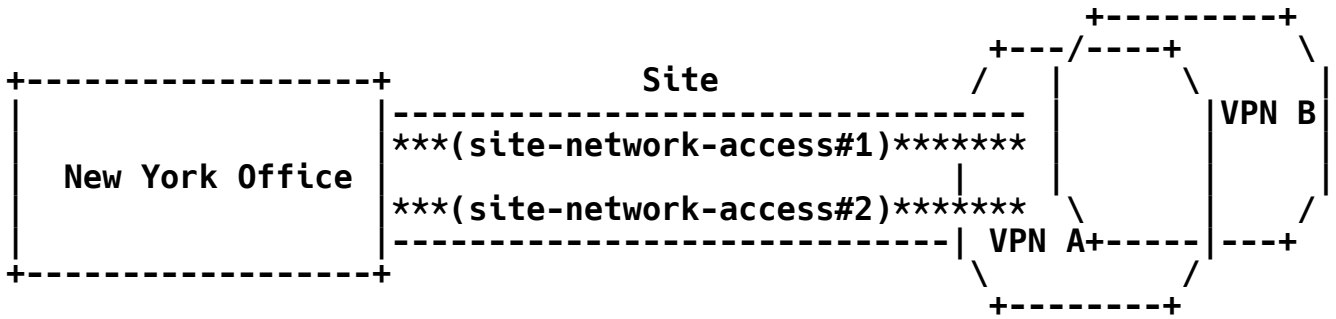
6.5.1.1. Single VPN Attachment: site-vpn-flavor-single

The figure below describes a single VPN attachment. The site connects to only one VPN.



6.5.1.2. MultiVPN Attachment: site-vpn-flavor-multi

The figure below describes a site connected to multiple VPNs.

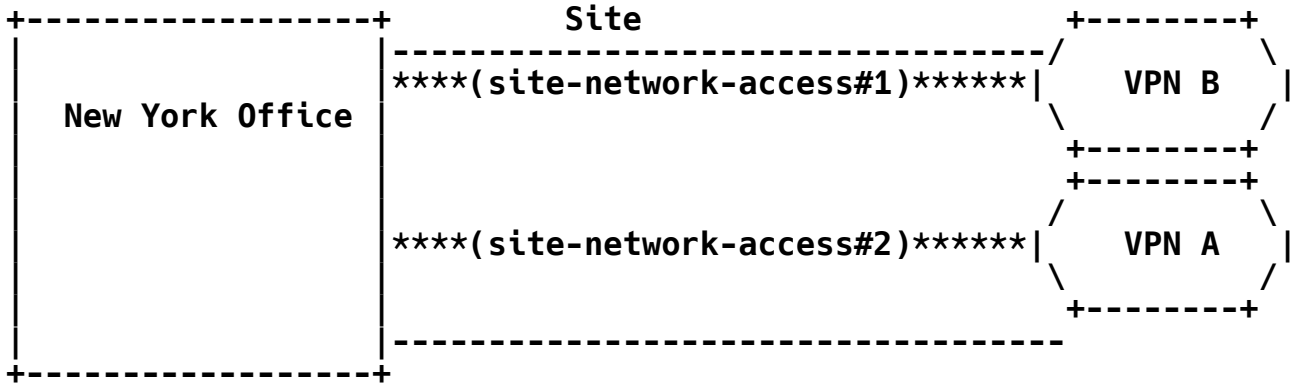


In the example above, the New York office is multihomed. Both logical accesses are using the same VPN attachment rules, and both are connected to VPN A and VPN B.

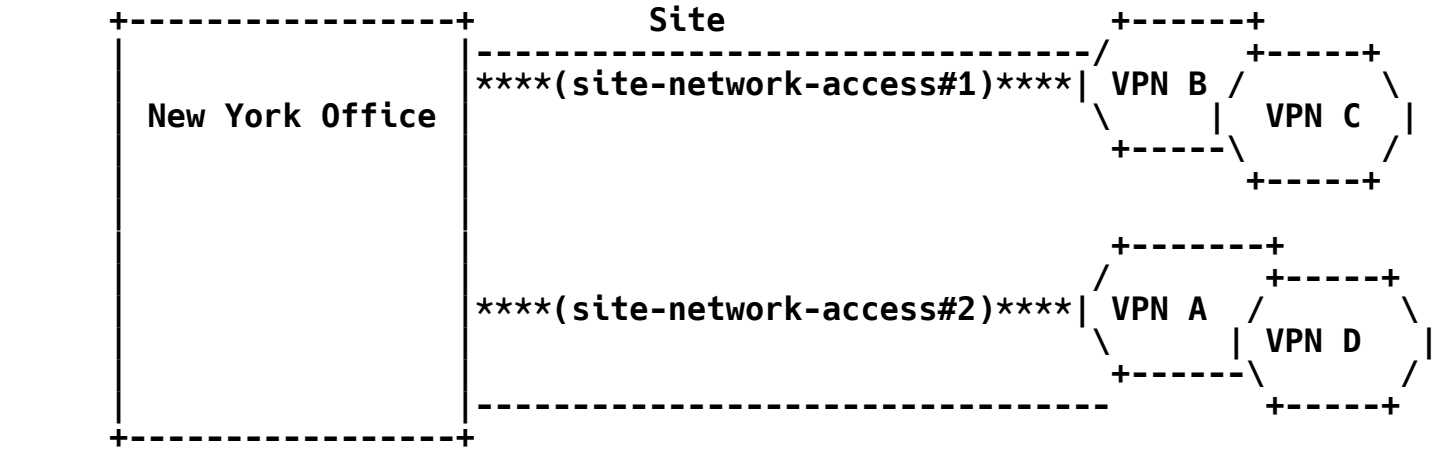
Reaching VPN A or VPN B from the New York office will be done via destination-based routing. Having the same destination reachable from the two VPNs may cause routing troubles. The customer administration's role in this case would be to ensure the appropriate mapping of its prefixes in each VPN.

6.5.1.3. SubVPN Attachment: site-vpn-flavor-sub

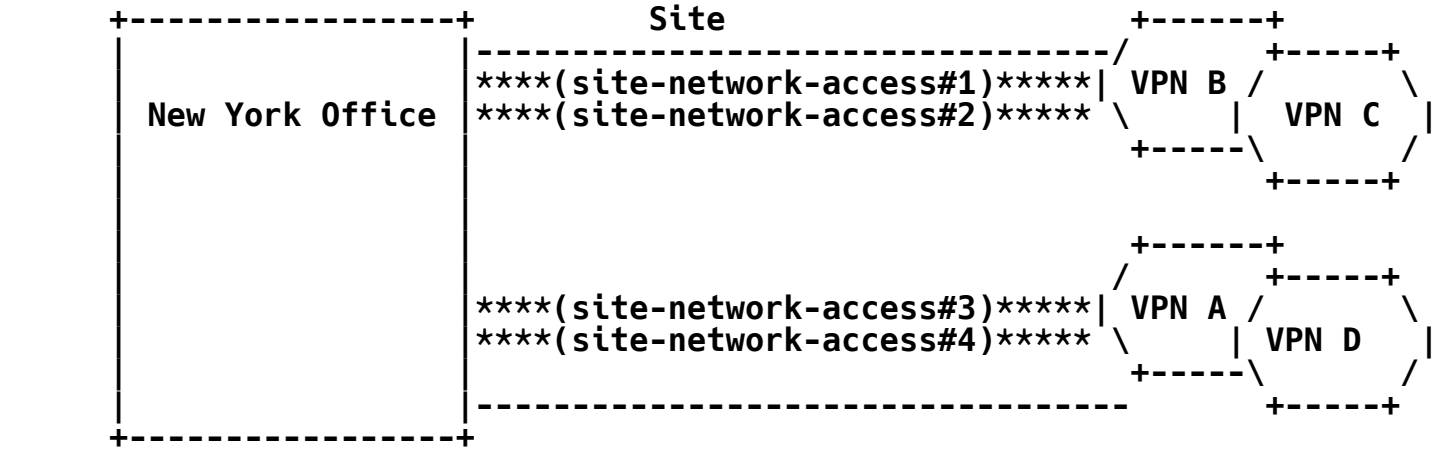
The figure below describes a subVPN attachment. The site connects to multiple VPNs, but each logical access is attached to a particular set of VPNs. A typical use case for a subVPN is a customer site used by multiple affiliates with private resources for each affiliate that cannot be shared (communication between the affiliates is prevented). It is similar to having separate sites, but in this case the customer wants to share some physical components while maintaining strong communication isolation between the affiliates. In this example, site-network-access#1 is attached to VPN B, while site-network-access#2 is attached to VPN A.



A multiVPN can be implemented in addition to a subVPN; as a consequence, each site-network-access can access multiple VPNs. In the example below, site-network-access#1 is mapped to VPN B and VPN C, while site-network-access#2 is mapped to VPN A and VPN D.



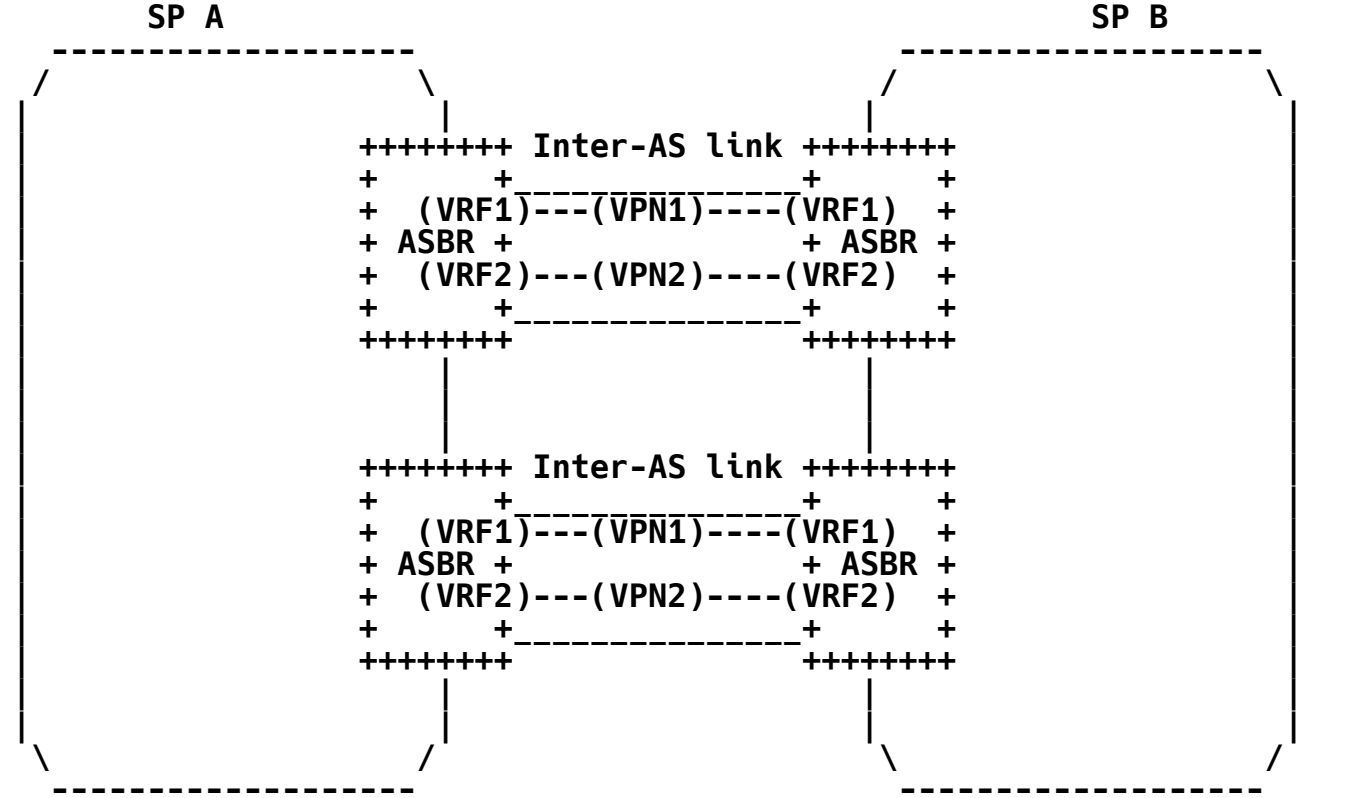
Multihoming is also possible with subVPNs; in this case, site-network-accesses are grouped, and a particular group will have access to the same set of VPNs. In the example below, site-network-access#1 and site-network-access#2 are part of the same group (multihomed together) and are mapped to VPN B and VPN C; in addition, site-network-access#3 and site-network-access#4 are part of the same group (multihomed together) and are mapped to VPN A and VPN D.



In terms of service configuration, a subVPN can be achieved by requesting that the site-network-access use the same bearer (see Sections 6.6.4 and 6.6.6.4 for more details).

6.5.1.4. NNI: site-vpn-flavor-nni

A Network-to-Network Interface (NNI) scenario may be modeled using the sites container (see Section 6.15.1). Using the sites container to model an NNI is only one possible option for NNIs (see Section 6.15). This option is called "option A" by reference to the option A NNI defined in [RFC4364]. It is helpful for the SP to indicate that the requested VPN connection is not a regular site but rather is an NNI, as specific default device configuration parameters may be applied in the case of NNIs (e.g., ACLs, routing policies).



The figure above describes an option A NNI scenario that can be modeled using the sites container. In order to connect its customer VPNs (VPN1 and VPN2) in SP B, SP A may request the creation of some site-network-accesses to SP B. The site-vpn-flavor-nni will be used to inform SP B that this is an NNI and not a regular customer site. The site-vpn-flavor-nni may be multihomed and multiVPN as well.

6.5.2. Attaching a Site to a VPN

Due to the multiple site-vpn flavors, the attachment of a site to an IP VPN is done at the site-network-access (logical access) level through the "vpn-attachment" container. The vpn-attachment container is mandatory. The model provides two ways to attach a site to a VPN:

- By referencing the target VPN directly.
- By referencing a VPN policy for attachments that are more complex.

A choice is implemented to allow the user to choose the flavor that provides the best fit.

6.5.2.1. Referencing a VPN

Referencing a vpn-id provides an easy way to attach a particular logical access to a VPN. This is the best way in the case of a single VPN attachment or subVPN with a single VPN attachment per logical access. When referencing a vpn-id, the site-role setting must be added to express the role of the site in the target VPN service topology.

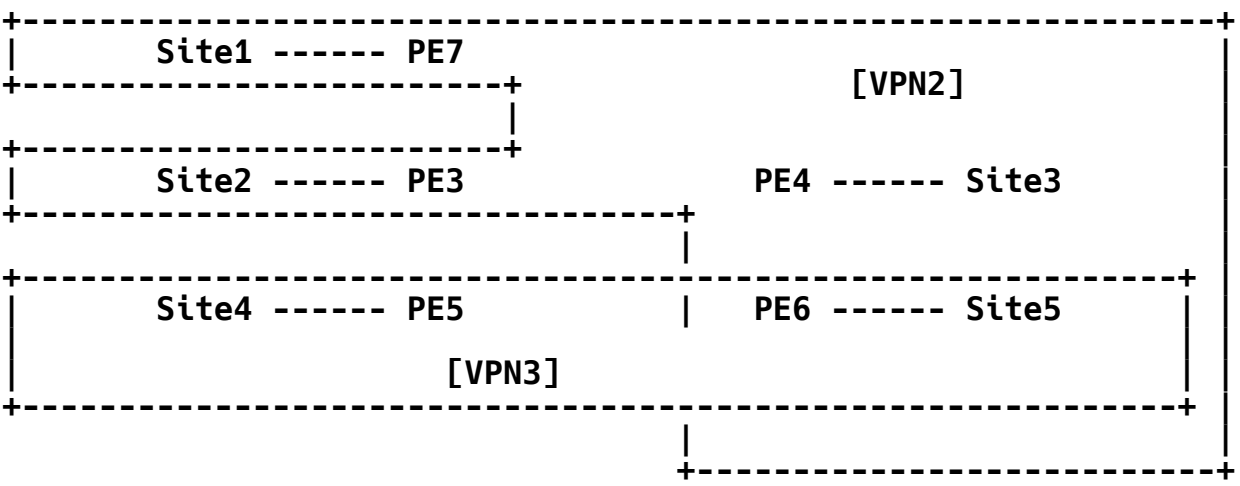
```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>LA1</site-network-access-id>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
  <site-network-access>
   <site-network-access-id>LA2</site-network-access-id>
   <vpn-attachment>
    <vpn-id>VPNB</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
</site>
```

The example above describes a subVPN case where a site (SITE1) has two logical accesses (LA1 and LA2), with LA1 attached to VPNA and LA2 attached to VPNB.

6.5.2.2. VPN Policy

The "vpn-policy" list helps express a multiVPN scenario where a logical access belongs to multiple VPNs. Multiple VPN policies can be created to handle the subVPN case where each logical access is part of a different set of VPNs.

As a site can belong to multiple VPNs, the vpn-policy list may be composed of multiple entries. A filter can be applied to specify that only some LANs of the site should be part of a particular VPN. Each time a site (or LAN) is attached to a VPN, the user must precisely describe its role (site-role) within the target VPN service topology.



In the example above, Site5 is part of two VPNs: VPN3 and VPN2. It will play a Hub role in VPN2 and an any-to-any role in VPN3. We can express such a multiVPN scenario as follows:

```
<site>
 <site-id>Site5</site-id>
 <vpn-policies>
  <vpn-policy>
   <vpn-policy-id>POLICY1</vpn-policy-id>
   <entries>
    <id>ENTRY1</id>
    <vpn>
     <vpn-id>VPN2</vpn-id>
     <site-role>hub-role</site-role>
    </vpn>
   </entries>
```

```
      <entries>
       <id>ENTRY2</id>
       <vpn>
        <vpn-id>VPN3</vpn-id>
        <site-role>any-to-any-role</site-role>
       </vpn>
      </entries>
     </vpn-policy>
    </vpn-policies>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>LA1</site-network-access-id>
      <vpn-attachment>
       <vpn-policy-id>POLICY1</vpn-policy-id>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

Now, if a more-granular VPN attachment is necessary, filtering can be used. For example, if LAN1 from Site5 must be attached to VPN2 as a Hub and LAN2 must be attached to VPN3, the following configuration can be used:

```
   <site>
    <site-id>Site5</site-id>
    <vpn-policies>
     <vpn-policy>
      <vpn-policy-id>POLICY1</vpn-policy-id>
      <entries>
       <id>ENTRY1</id>
       <filter>
        <lan-tag>LAN1</lan-tag>
       </filter>
       <vpn>
        <vpn-id>VPN2</vpn-id>
        <site-role>hub-role</site-role>
       </vpn>
      </entries>
      <entries>
       <id>ENTRY2</id>
       <filter>
        <lan-tag>LAN2</lan-tag>
       </filter>
```

```
       <vpn>
        <vpn-id>VPN3</vpn-id>
        <site-role>any-to-any-role</site-role>
       </vpn>
      </entries>
     </vpn-policy>
    </vpn-policies>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>LA1</site-network-access-id>
      <vpn-attachment>
       <vpn-policy-id>POLICY1</vpn-policy-id>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

6.6. Deciding Where to Connect the Site

The management system will have to determine where to connect each site-network-access of a particular site to the provider network (e.g., PE, aggregation switch).

The current model proposes parameters and constraints that can influence the meshing of the site-network-access.

The management system SHOULD honor any customer constraints. If a constraint is too strict and cannot be fulfilled, the management system MUST NOT provision the site and SHOULD provide relevant information to the user. How the information is provided is out of scope for this document. Whether or not to relax the constraint would then be left up to the user.

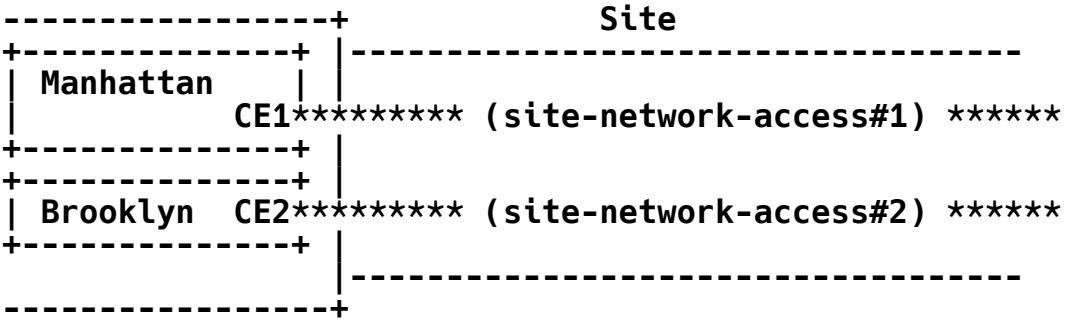
Parameters are just hints for the management system for service placement.

In addition to parameters and constraints, the management system's decision MAY be based on any other internal constraints that are left up to the SP: least load, distance, etc.

6.6.1. Constraint: Device

In the case of provider management or co-management, one or more devices have been ordered by the customer. The customer may force a particular site-network-access to be connected on a particular device that he ordered.

New York Site

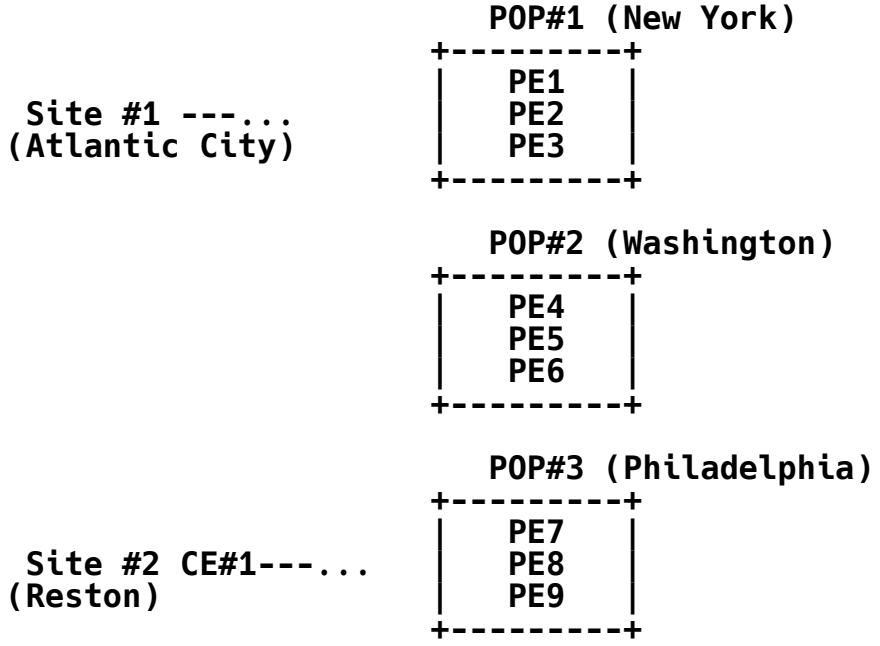


In the figure above, site-network-access#1 is associated with CE1 in the service request. The SP must ensure the provisioning of this connection.

6.6.2. Constraint/Parameter: Site Location

The location information provided in this model MAY be used by a management system to determine the target PE to mesh the site (SP side). A particular location must be associated with each site network access when configuring it. The SP MUST honor the termination of the access on the location associated with the site network access (customer side). The "country-code" in the site location SHOULD be expressed as an ISO ALPHA-2 code.

The site-network-access location is determined by the "location-flavor". In the case of a provider-managed or co-managed site, the user is expected to configure a "device-reference" (device case) that will bind the site-network-access to a particular device that the customer ordered. As each device is already associated with a particular location, in such a case the location information is retrieved from the device location. In the case of a customer-managed site, the user is expected to configure a "location-reference" (location case); this provides a reference to an existing configured location and will help with placement.



In the example above, Site #1 is a customer-managed site with a location L1, while Site #2 is a provider-managed site for which a CE (CE#1) was ordered. Site #2 is configured with L2 as its location. When configuring a site-network-access for Site #1, the user will need to reference location L1 so that the management system will know that the access will need to terminate on this location. Then, for distance reasons, this management system may mesh Site #1 on a PE in the Philadelphia POP. It may also take into account resources available on PEs to determine the exact target PE (e.g., least loaded). For Site #2, the user is expected to configure the site-network-access with a device-reference to CE#1 so that the management system will know that the access must terminate on the location of CE#1 and must be connected to CE#1. For placement of the SP side of the access connection, in the case of the nearest PE used, it may mesh Site #2 on the Washington POP.

6.6.3. Constraint/Parameter: Access Type

The management system needs to elect the access media to connect the site to the customer (for example, xDSL, leased line, Ethernet backhaul). The customer may provide some parameters/constraints that will provide hints to the management system.

The bearer container information SHOULD be the first piece of information considered when making this decision:

- The "requested-type" parameter provides information about the media type that the customer would like to use. If the "strict" leaf is equal to "true", this MUST be considered a strict constraint so that the management system cannot connect the site with another media type. If the "strict" leaf is equal to "false" (default) and if the requested media type cannot be fulfilled, the management system can select another media type. The supported media types SHOULD be communicated by the SP to the customer via a mechanism that is out of scope for this document.

- The "always-on" leaf defines a strict constraint: if set to true, the management system MUST elect a media type that is "always-on" (e.g., this means no dial access type).

- The "bearer-reference" parameter is used in cases where the customer has already ordered a network connection to the SP apart from the IP VPN site and wants to reuse this connection. The string used is an internal reference from the SP and describes the already-available connection. This is also a strict requirement that cannot be relaxed. How the reference is given to the customer is out of scope for this document, but as a pure example, when the customer ordered the bearer (through a process that is out of scope for this model), the SP may have provided the bearer reference that can be used for provisioning services on top.

Any other internal parameters from the SP can also be used. The management system MAY use other parameters, such as the requested "svc-input-bandwidth" and "svc-output-bandwidth", to help decide which access type to use.

6.6.4. Constraint: Access Diversity

Each site-network-access may have one or more constraints that would drive the placement of the access. By default, the model assumes that there are no constraints, but allocation of a unique bearer per site-network-access is expected.

In order to help with the different placement scenarios, a site-network-access may be tagged using one or multiple group identifiers. The group identifier is a string, so it can accommodate both explicit naming of a group of sites (e.g., "multihomed-set1" or "subVPN") and the use of a numbered identifier (e.g., 12345678). The meaning of each group-id is local to each customer administrator, and the management system MUST ensure that different customers can use the same group-ids. One or more group-ids can also be defined at the

site level; as a consequence, all site-network-accesses under the site MUST inherit the group-ids of the site they belong to. When, in addition to the site group-ids some group-ids are defined at the site-network-access level, the management system MUST consider the union of all groups (site level and site network access level) for this particular site-network-access.

For an already-configured site-network-access, each constraint MUST be expressed against a targeted set of site-network-accesses. This site-network-access MUST never be taken into account in the targeted set -- for example, "My site-network-access S must not be connected on the same POP as the site-network-accesses that are part of Group 10." The set of site-network-accesses against which the constraint is evaluated can be expressed as a list of groups, "all-other-accesses", or "all-other-groups". The all-other-accesses option means that the current site-network-access constraint MUST be evaluated against all the other site-network-accesses belonging to the current site. The all-other-groups option means that the constraint MUST be evaluated against all groups that the current site-network-access does not belong to.

The current model proposes multiple constraint-types:

- pe-diverse: The current site-network-access MUST NOT be connected to the same PE as the targeted site-network-accesses.
- pop-diverse: The current site-network-access MUST NOT be connected to the same POP as the targeted site-network-accesses.
- linecard-diverse: The current site-network-access MUST NOT be connected to the same linecard as the targeted site-network-accesses.
- bearer-diverse: The current site-network-access MUST NOT use common bearer components compared to bearers used by the targeted site-network-accesses. "bearer-diverse" provides some level of diversity at the access level. As an example, two bearer-diverse site-network-accesses must not use the same DSLAM, BAS, or Layer 2 switch.
- same-pe: The current site-network-access MUST be connected to the same PE as the targeted site-network-accesses.
- same-bearer: The current site-network-access MUST be connected using the same bearer as the targeted site-network-accesses.

These constraint-types can be extended through augmentation.

Each constraint is expressed as "The site-network-access S must be <constraint-type> (e.g., pe-diverse, pop-diverse) from these <target> site-network-accesses."

The group-id used to target some site-network-accesses may be the same as the one used by the current site-network-access. This eases the configuration of scenarios where a group of site-network-access points has a constraint between the access points in the group. As an example, if we want a set of sites (Site#1 to Site#5) to be connected on different PEs, we can tag them with the same group-id and express a pe-diverse constraint for this group-id.

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>10</group-id>
     </group>
    </groups>
    <constraints>
     <constraint>
      <constraint-type>pe-diverse</constraint-type>
      <target>
       <group>
        <group-id>10</group-id>
       </group>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
</site>
```

```
   <site>
    <site-id>SITE2</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>10</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pe-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   ...
   <site>
    <site-id>SITE5</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>10</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pe-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
```

```
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNA</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
   </site-network-accesses>
  </site>
```

The group-id used to target some site-network-accesses may also be different than the one used by the current site-network-access. This can be used to express that a group of sites has some constraints against another group of sites, but there is no constraint within the group. For example, we consider a set of six sites and two groups; we want to ensure that a site in the first group must be pop-diverse from a site in the second group:

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>10</group-id>
     </group>
    </groups>
    <constraints>
     <constraint>
      <constraint-type>pop-diverse</constraint-type>
      <target>
       <group>
        <group-id>20</group-id>
       </group>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
```

```
   </site>
   <site>
    <site-id>SITE2</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>10</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>20</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   ...
   <site>
    <site-id>SITE5</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
```

```
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   <site>
    <site-id>SITE6</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

6.6.5. Infeasible Access Placement

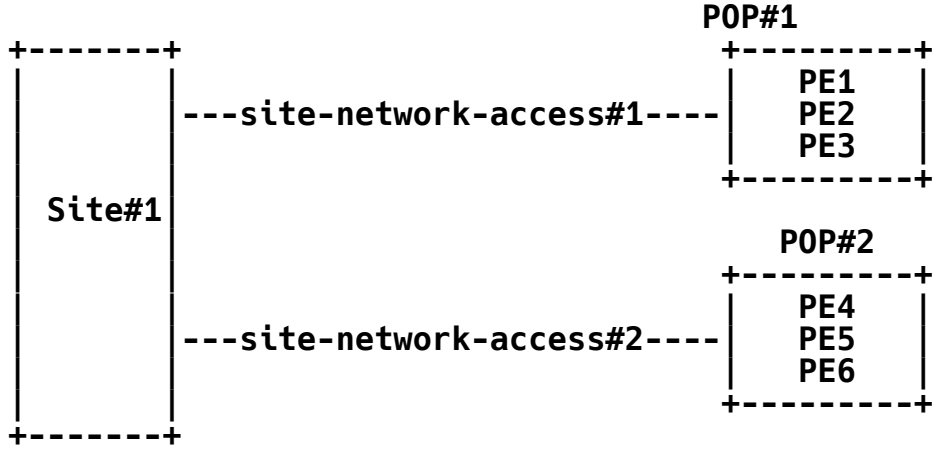
Some infeasible access placement scenarios could be created via the proposed configuration framework. Such infeasible access placement scenarios could result from constraints that are too restrictive, leading to infeasible access placement in the network or conflicting

constraints that would also lead to infeasible access placement. An example of conflicting rules would be to request that site-network-access#1 be pe-diverse from site-network-access#2 and to request at the same time that site-network-access#2 be on the same PE as site-network-access#1. When the management system cannot determine the placement of a site-network-access, it SHOULD return an error message indicating that placement was not possible.

6.6.6. Examples of Access Placement

6.6.6.1. Multihoming

The customer wants to create a multihomed site. The site will be composed of two site-network-accesses; for resiliency purposes, the customer wants the two site-network-accesses to be meshed on different POPs.



This scenario can be expressed as follows:

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>10</group-id>
     </group>
    </groups>
    <constraints>
     <constraint>
      <constraint-type>pop-diverse</constraint-type>
```

```
        <target>
         <group>
          <group-id>20</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNA</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
    <site-network-access>
     <site-network-access-id>2</site-network-access-id>
     <access-diversity>
      <groups>
       <group>
        <group-id>20</group-id>
       </group>
      </groups>
      <constraints>
       <constraint>
        <constraint-type>pop-diverse</constraint-type>
        <target>
         <group>
          <group-id>10</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNA</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
   </site-network-accesses>
  </site>
```

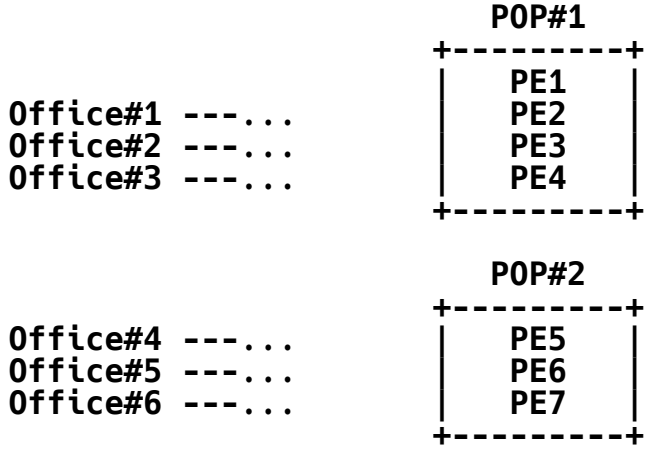
But it can also be expressed as follows:

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <constraints>
     <constraint>
      <constraint-type>pop-diverse</constraint-type>
      <target>
       <all-other-accesses/>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
  <site-network-access>
   <site-network-access-id>2</site-network-access-id>
   <access-diversity>
    <constraints>
     <constraint>
      <constraint-type>pop-diverse</constraint-type>
      <target>
       <all-other-accesses/>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNA</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
</site>
```

6.6.6.2. Site Offload

The customer has six branch offices in a particular region, and he wants to prevent having all branch offices connected on the same PE.

He wants to express that three branch offices cannot be connected on the same linecard. Also, the other branch offices must be connected on a different POP. Those other branch offices cannot also be connected on the same linecard.



This scenario can be expressed as follows:

- We need to create two groups of sites: Group#10, which is composed of Office#1, Office#2, and Office#3; and Group#20, which is composed of Office#4, Office#5, and Office#6.
- Sites within Group#10 must be pop-diverse from sites within Group#20, and vice versa.
- Sites within Group#10 must be linecard-diverse from other sites in Group#10 (same for Group#20).

```
   <site>
    <site-id>Office1</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>10</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>20</group-id>
          </group>
         </target>
        </constraint>
        <constraint>
         <constraint-type>linecard-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   <site>
    <site-id>Office2</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>10</group-id>
        </group>
       </groups>
```

```
      <constraints>
       <constraint>
        <constraint-type>pop-diverse</constraint-type>
        <target>
         <group>
          <group-id>20</group-id>
         </group>
        </target>
       </constraint>
       <constraint>
        <constraint-type>linecard-diverse</constraint-type>
        <target>
         <group>
          <group-id>10</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNA</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
   </site-network-accesses>
  </site>
  <site>
   <site-id>Office3</site-id>
   <site-network-accesses>
    <site-network-access>
     <site-network-access-id>1</site-network-access-id>
     <access-diversity>
      <groups>
       <group>
        <group-id>10</group-id>
       </group>
      </groups>
      <constraints>
       <constraint>
        <constraint-type>pop-diverse</constraint-type>
        <target>
         <group>
          <group-id>20</group-id>
         </group>
        </target>
       </constraint>
```

```
        <constraint>
         <constraint-type>linecard-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   <site>
    <site-id>Office4</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
        <constraint>
         <constraint-type>linecard-diverse</constraint-type>
         <target>
          <group>
           <group-id>20</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
```

```
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
   <site>
    <site-id>Office5</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
        <constraint>
         <constraint-type>linecard-diverse</constraint-type>
         <target>
          <group>
           <group-id>20</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

```
   <site>
    <site-id>Office6</site-id>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pop-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
        <constraint>
         <constraint-type>linecard-diverse</constraint-type>
         <target>
          <group>
           <group-id>20</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNA</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

6.6.6.3. Parallel Links

To increase its site bandwidth at lower cost, a customer wants to order two parallel site-network-accesses that will be connected to the same PE.

```
       *******site-network-access#1**********
Site 1 *******site-network-access#2********** PE1
```

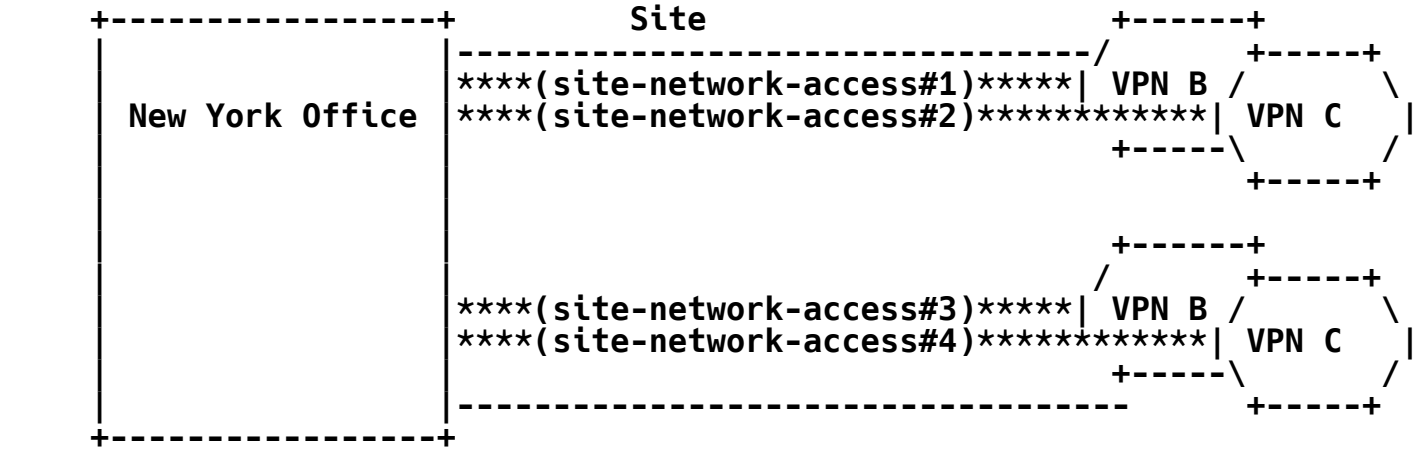
This scenario can be expressed as follows:

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>PE-linkgrp-1</group-id>
     </group>
    </groups>
    <constraints>
     <constraint>
      <constraint-type>same-pe</constraint-type>
      <target>
       <group>
        <group-id>PE-linkgrp-1</group-id>
       </group>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNB</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
  <site-network-access>
   <site-network-access-id>2</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>PE-linkgrp-1</group-id>
     </group>
    </groups>
```

```
       <constraints>
        <constraint>
         <constraint-type>same-pe</constraint-type>
         <target>
          <group>
           <group-id>PE-linkgrp-1</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <vpn-attachment>
       <vpn-id>VPNB</vpn-id>
       <site-role>spoke-role</site-role>
      </vpn-attachment>
     </site-network-access>
    </site-network-accesses>
   </site>
```

6.6.6.4. SubVPN with Multihoming

A customer has a site that is dual-homed. The dual-homing must be done on two different PEs. The customer also wants to implement two subVPNs on those multihomed accesses.



This scenario can be expressed as follows:

- The site will have four site network accesses (two subVPNs coupled via dual-homing).
- Site-network-access#1 and site-network-access#3 will correspond to the multihoming of subVPN B. A PE-diverse constraint is required between them.

- Site-network-access#2 and site-network-access#4 will correspond to the multihoming of subVPN C. A PE-diverse constraint is required between them.

- To ensure proper usage of the same bearer for the subVPN, site-network-access#1 and site-network-access#2 must share the same bearer as site-network-access#3 and site-network-access#4.

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
      <group-id>dualhomed-1</group-id>
     </group>
    </groups>
    <constraints>
     <constraint>
      <constraint-type>pe-diverse</constraint-type>
      <target>
       <group>
        <group-id>dualhomed-2</group-id>
       </group>
      </target>
     </constraint>
     <constraint>
      <constraint-type>same-bearer</constraint-type>
      <target>
       <group>
        <group-id>dualhomed-1</group-id>
       </group>
      </target>
     </constraint>
    </constraints>
   </access-diversity>
   <vpn-attachment>
    <vpn-id>VPNB</vpn-id>
    <site-role>spoke-role</site-role>
   </vpn-attachment>
  </site-network-access>
  <site-network-access>
   <site-network-access-id>2</site-network-access-id>
   <access-diversity>
    <groups>
     <group>
```

```
        <group-id>dualhomed-1</group-id>
       </group>
      </groups>
      <constraints>
       <constraint>
        <constraint-type>pe-diverse</constraint-type>
        <target>
         <group>
          <group-id>dualhomed-2</group-id>
         </group>
        </target>
       </constraint>
       <constraint>
        <constraint-type>same-bearer</constraint-type>
        <target>
         <group>
          <group-id>dualhomed-1</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNC</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
    <site-network-access>
     <site-network-access-id>3</site-network-access-id>
     <access-diversity>
      <groups>
       <group>
        <group-id>dualhomed-2</group-id>
       </group>
      </groups>
      <constraints>
       <constraint>
        <constraint-type>pe-diverse</constraint-type>
        <target>
         <group>
          <group-id>dualhomed-1</group-id>
         </group>
        </target>
       </constraint>
       <constraint>
        <constraint-type>same-bearer</constraint-type>
        <target>
         <group>
```

```
          <group-id>dualhomed-2</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNB</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
    <site-network-access>
     <site-network-access-id>4</site-network-access-id>
     <access-diversity>
      <groups>
       <group>
        <group-id>dualhomed-2</group-id>
       </group>
      </groups>
      <constraints>
       <constraint>
        <constraint-type>pe-diverse</constraint-type>
        <target>
         <group>
          <group-id>dualhomed-1</group-id>
         </group>
        </target>
       </constraint>
       <constraint>
        <constraint-type>same-bearer</constraint-type>
        <target>
         <group>
          <group-id>dualhomed-2</group-id>
         </group>
        </target>
       </constraint>
      </constraints>
     </access-diversity>
     <vpn-attachment>
      <vpn-id>VPNC</vpn-id>
      <site-role>spoke-role</site-role>
     </vpn-attachment>
    </site-network-access>
   </site-network-accesses>
  </site>
```

6.6.7. Route Distinguisher and VRF Allocation

The route distinguisher (RD) is a critical parameter of PE-based L3VPNs as described in [RFC4364] that provides the ability to distinguish common addressing plans in different VPNs. As for route targets (RTs), a management system is expected to allocate a VRF on the target PE and an RD for this VRF.

If a VRF already exists on the target PE and the VRF fulfills the connectivity constraints for the site, there is no need to recreate another VRF, and the site MAY be meshed within this existing VRF. How the management system checks that an existing VRF fulfills the connectivity constraints for a site is out of scope for this document.

If no such VRF exists on the target PE, the management system has to initiate the creation of a new VRF on the target PE and has to allocate a new RD for this new VRF.

The management system MAY apply a per-VPN or per-VRF allocation policy for the RD, depending on the SP's policy. In a per-VPN allocation policy, all VRFs (dispatched on multiple PEs) within a VPN will share the same RD value. In a per-VRF model, all VRFs should always have a unique RD value. Some other allocation policies are also possible, and this document does not restrict the allocation policies to be used.

The allocation of RDs MAY be done in the same way as RTs. The examples provided in Section 6.2.1.1 could be reused in this scenario.

Note that an SP MAY configure a target PE for an automated allocation of RDs. In this case, there will be no need for any backend system to allocate an RD value.

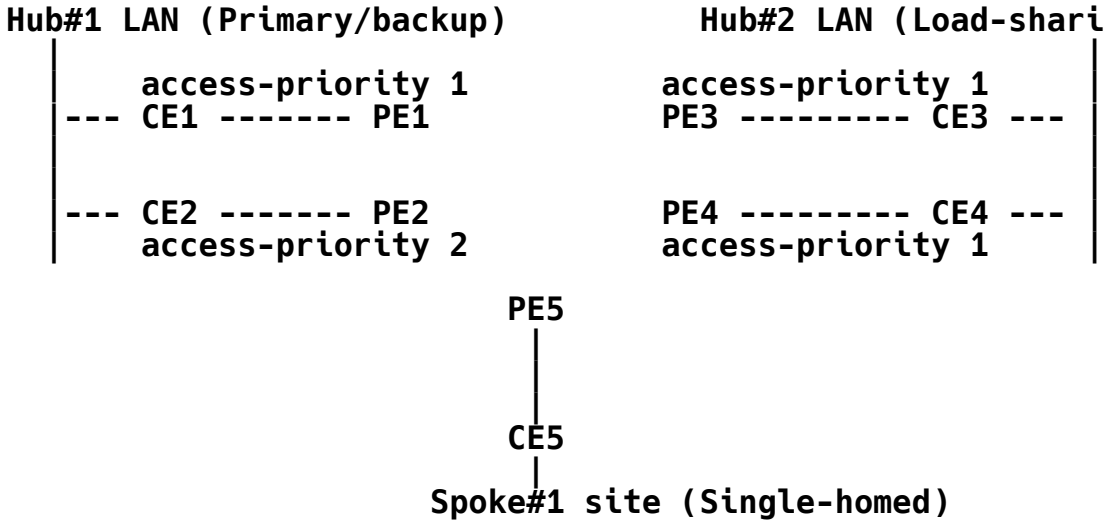
6.7. Site Network Access Availability

A site may be multihomed, meaning that it has multiple site-network-access points. Placement constraints defined in previous sections will help ensure physical diversity.

When the site-network-accesses are placed on the network, a customer may want to use a particular routing policy on those accesses.

The "site-network-access/availability" container defines parameters for site redundancy. The "access-priority" leaf defines a preference for a particular access. This preference is used to model load-balancing or primary/backup scenarios. The higher the access-priority value, the higher the preference will be.

The figure below describes how the access-priority attribute can be used.



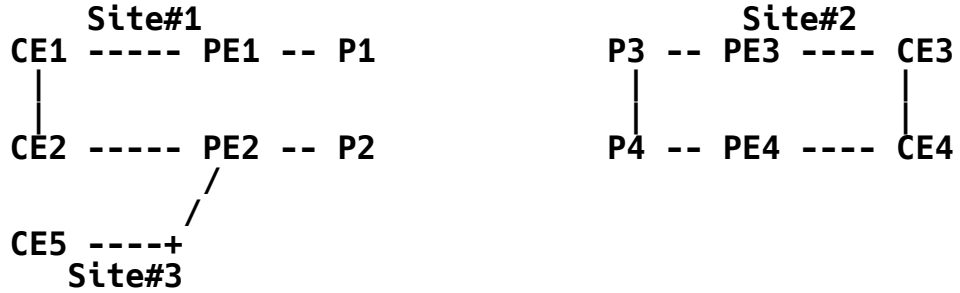
In the figure above, Hub#2 requires load-sharing, so all the site-network-accesses must use the same access-priority value. On the other hand, as Hub#1 requires a primary site-network-access and a backup site-network-access, a higher access-priority setting will be configured on the primary site-network-access.

Scenarios that are more complex can be modeled. Let's consider a Hub site with five accesses to the network (A1,A2,A3,A4,A5). The customer wants to load-share its traffic on A1,A2 in the nominal situation. If A1 and A2 fail, the customer wants to load-share its traffic on A3 and A4; finally, if A1 to A4 are down, he wants to use A5. We can model this easily by configuring the following access-priority values: A1=100, A2=100, A3=50, A4=50, A5=10.

The access-priority scenario has some limitations. An access-priority scenario like the previous one with five accesses but with the constraint of having traffic load-shared between A3 and A4 in the case where A1 OR A2 is down is not achievable. But the authors believe that using the access-priority attribute will cover most of the deployment use cases and that the model can still be extended via augmentation to support additional use cases.

6.8. Traffic Protection

The service model supports the ability to protect the traffic for a site. Such protection provides a better level of availability in multihoming scenarios by, for example, using local-repair techniques in case of failures. The associated level of service guarantee would be based on an agreement between the customer and the SP and is out of scope for this document.



In the figure above, we consider an IP VPN service with three sites, including two dual-homed sites (Site#1 and Site#2). For dual-homed sites, we consider PE1-CE1 and PE3-CE3 as primary and PE2-CE2,PE4-CE4 as backup for the example (even if protection also applies to load-sharing scenarios).

In order to protect Site#2 against a failure, a user may set the "traffic-protection/enabled" leaf to true for Site#2. How the traffic protection will be implemented is out of scope for this document. However, in such a case, we could consider traffic coming from a remote site (Site#1 or Site#3), where the primary path would use PE3 as the egress PE. PE3 may have preprogrammed a backup forwarding entry pointing to the backup path (through PE4-CE4) for all prefixes going through the PE3-CE3 link. How the backup path is computed is out of scope for this document. When the PE3-CE3 link fails, traffic is still received by PE3, but PE3 automatically switches traffic to the backup entry; the path will therefore be PE1-P1-(...)-P3-PE3-PE4-CE4 until the remote PEs reconverge and use PE4 as the egress PE.

6.9. Security

The "security" container defines customer-specific security parameters for the site. The security options supported in the model are limited but may be extended via augmentation.

6.9.1. Authentication

The current model does not support any authentication parameters for the site connection, but such parameters may be added in the "authentication" container through augmentation.

6.9.2. Encryption

Traffic encryption can be requested on the connection. It may be performed at Layer 2 or Layer 3 by selecting the appropriate enumeration in the "layer" leaf. For example, an SP may use IPsec when a customer requests Layer 3 encryption. The encryption profile can be SP defined or customer specific.

When an SP profile is used and a key (e.g., a pre-shared key) is allocated by the provider to be used by a customer, the SP should provide a way to communicate the key in a secured way to the customer.

When a customer profile is used, the model supports only a pre-shared key for authentication, with the pre-shared key provided through the NETCONF or RESTCONF request. A secure channel must be used to ensure that the pre-shared key cannot be intercepted.

For security reasons, it may be necessary for the customer to change the pre-shared key on a regular basis. To perform a key change, the user can ask the SP to change the pre-shared key by submitting a new pre-shared key for the site configuration (as shown below). This mechanism might not be hitless.

```
<site>
 <site-id>SITE1</site-id>
 <site-network-accesses>
  <site-network-access>
   <site-network-access-id>1</site-network-access-id>
   <security>
    <encryption-profile>
     <preshared-key>MY_NEW_KEY</preshared-key>
    </encryption-profile>
   </security>
  </site-network-access>
 </site-network-accesses>
</site>
```

A hitless key-change mechanism may be added through augmentation.

Other key-management methodologies may be added through augmentation. A "pki" container, which is empty, has been created to help with support of PKI through augmentation.

6.10. Management

The model proposes three types of common management options:

- provider-managed: The CE router is managed only by the provider. In this model, the responsibility boundary between the SP and the customer is between the CE and the customer network.

- customer-managed: The CE router is managed only by the customer. In this model, the responsibility boundary between the SP and the customer is between the PE and the CE.

- co-managed: The CE router is primarily managed by the provider; in addition, the SP allows customers to access the CE for configuration/monitoring purposes. In the co-managed mode, the responsibility boundary is the same as the responsibility boundary for the provider-managed model.

Based on the management model, different security options MAY be derived.

In the co-managed case, the model proposes some options to define the management address family (IPv4 or IPv6) and the associated management address.

6.11. Routing Protocols

"routing-protocol" defines which routing protocol must be activated between the provider and the customer router. The current model supports the following settings: bgp, rip, ospf, static, direct, and vrrp.

The routing protocol defined applies at the provider-to-customer boundary. Depending on how the management model is administered, it may apply to the PE-CE boundary or the CE-to-customer boundary. In the case of a customer-managed site, the routing protocol defined will be activated between the PE and the CE router managed by the customer. In the case of a provider-managed site, the routing protocol defined will be activated between the CE managed by the SP and the router or LAN belonging to the customer. In this case, we expect the PE-CE routing to be configured based on the SP's rules, as both are managed by the same entity.

```
                              Rtg protocol
      192.0.2.0/24 ----- CE ----------------- PE1

                   Customer-managed site

            Rtg protocol
      Customer router ----- CE ----------------- PE1

                   Provider-managed site
```

All the examples below will refer to a scenario for a customer-managed site.

6.11.1. Handling of Dual Stack

All routing protocol types support dual stack by using the "address-family" leaf-list.

Example of dual stack using the same routing protocol:

```
<routing-protocols>
  <routing-protocol>
    <type>static</type>
    <static>
        <address-family>ipv4</address-family>
        <address-family>ipv6</address-family>
    </static>
  </routing-protocol>
</routing-protocols>
```

Example of dual stack using two different routing protocols:

```
<routing-protocols>
  <routing-protocol>
    <type>rip</type>
    <rip>
        <address-family>ipv4</address-family>
    </rip>
  </routing-protocol>
  <routing-protocol>
    <type>ospf</type>
    <ospf>
        <address-family>ipv6</address-family>
    </ospf>
  </routing-protocol>
</routing-protocols>
```

6.11.2. LAN Directly Connected to SP Network

The routing protocol type "direct" SHOULD be used when a customer LAN is directly connected to the provider network and must be advertised in the IP VPN.

LAN attached directly to provider network:

192.0.2.0/24 ----- PE1

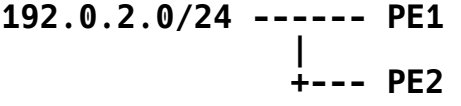
In this case, the customer has a default route to the PE address.

6.11.3. LAN Directly Connected to SP Network with Redundancy

The routing protocol type "vrrp" SHOULD be used and advertised in the IP VPN when

- the customer LAN is directly connected to the provider network, and
- LAN redundancy is expected.

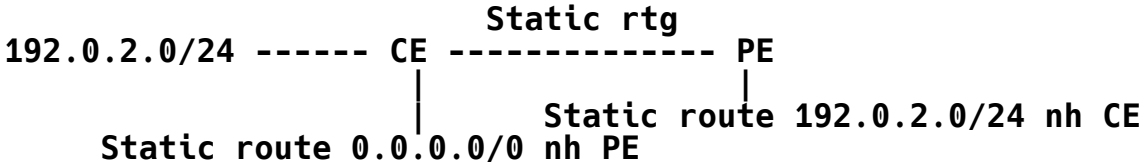
LAN attached directly to provider network with LAN redundancy:



In this case, the customer has a default route to the SP network.

6.11.4. Static Routing

The routing protocol type "static" MAY be used when a customer LAN is connected to the provider network through a CE router and must be advertised in the IP VPN. In this case, the static routes give next hops (nh) to the CE and to the PE. The customer has a default route to the SP network.



6.11.5. RIP Routing

The routing protocol type "rip" MAY be used when a customer LAN is connected to the provider network through a CE router and must be advertised in the IP VPN. For IPv4, the model assumes that RIP version 2 is used.

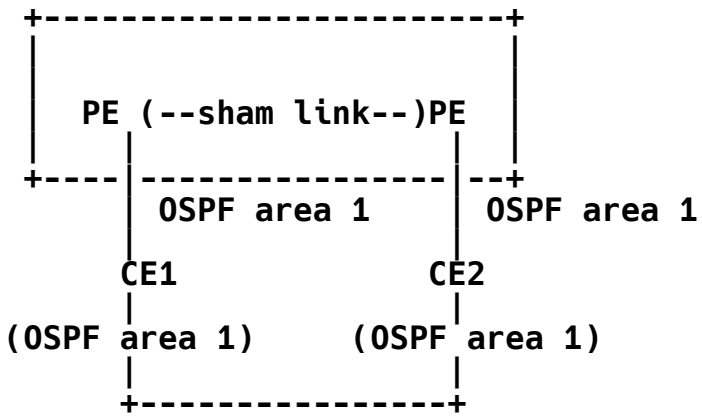
In the case of dual-stack routing requested through this model, the management system will be responsible for configuring RIP (including the correct version number) and associated address families on network elements.

6.11.6. OSPF Routing

The routing protocol type "ospf" MAY be used when a customer LAN is connected to the provider network through a CE router and must be advertised in the IP VPN.

It can be used to extend an existing OSPF network and interconnect different areas. See [RFC4577] for more details.

The model also proposes an option to create an OSPF sham link between two sites sharing the same area and having a backdoor link. The sham link is created by referencing the target site sharing the same OSPF area. The management system will be responsible for checking to see if there is already a sham link configured for this VPN and area between the same pair of PEs. If there is no existing sham link, the management system will provision one. This sham link MAY be reused by other sites.



Regarding dual-stack support, the user MAY specify both IPv4 and IPv6 address families, if both protocols should be routed through OSPF. As OSPF uses separate protocol instances for IPv4 and IPv6, the management system will need to configure both OSPF version 2 and OSPF version 3 on the PE-CE link.

Example of OSPF routing parameters in the service model:

```
<routing-protocols>
  <routing-protocol>
    <type>ospf</type>
    <ospf>
        <area-address>0.0.0.1</area-address>
        <address-family>ipv4</address-family>
        <address-family>ipv6</address-family>
    </ospf>
  </routing-protocol>
</routing-protocols>
```

Example of PE configuration done by the management system:

```
router ospf 10
 area 0.0.0.1
  interface Ethernet0/0
!
router ospfv3 10
 area 0.0.0.1
  interface Ethernet0/0
 !
```

6.11.7. BGP Routing

The routing protocol type "bgp" MAY be used when a customer LAN is connected to the provider network through a CE router and must be advertised in the IP VPN.

The session addressing will be derived from connection parameters as well as the SP's knowledge of the addressing plan that is in use.

In the case of dual-stack access, the user MAY request BGP routing for both IPv4 and IPv6 by specifying both address families. It will be up to the SP and management system to determine how to decline the configuration (two BGP sessions, single, multi-session, etc.).

The service configuration below activates BGP on the PE-CE link for both IPv4 and IPv6.

BGP activation requires the SP to know the address of the customer peer. The "static-address" allocation type for the IP connection MUST be used.

```
<routing-protocols>
  <routing-protocol>
    <type>bgp</type>
    <bgp>
        <autonomous-system>65000</autonomous-system>
        <address-family>ipv4</address-family>
        <address-family>ipv6</address-family>
    </bgp>
  </routing-protocol>
</routing-protocols>
```

Depending on the SP flavor, a management system can divide this service configuration into different flavors, as shown by the following examples.

Example of PE configuration done by the management system (single IPv4 transport session):

```
router bgp 100
 neighbor 203.0.113.2 remote-as 65000
 address-family ipv4 vrf Cust1
    neighbor 203.0.113.2 activate
 address-family ipv6 vrf Cust1
    neighbor 203.0.113.2 activate
    neighbor 203.0.113.2 route-map SET-NH-IPV6 out
```

Example of PE configuration done by the management system (two sessions):

```
router bgp 100
 neighbor 203.0.113.2 remote-as 65000
 neighbor 2001::2 remote-as 65000
 address-family ipv4 vrf Cust1
    neighbor 203.0.113.2 activate
 address-family ipv6 vrf Cust1
    neighbor 2001::2 activate
```

Example of PE configuration done by the management system (multi-session):

```
router bgp 100
 neighbor 203.0.113.2 remote-as 65000
 neighbor 203.0.113.2 multisession per-af
 address-family ipv4 vrf Cust1
    neighbor 203.0.113.2 activate
 address-family ipv6 vrf Cust1
    neighbor 203.0.113.2 activate
    neighbor 203.0.113.2 route-map SET-NH-IPV6 out
```

6.12. Service

The service defines service parameters associated with the site.

6.12.1. Bandwidth

The service bandwidth refers to the bandwidth requirement between the PE and the CE (WAN link bandwidth). The requested bandwidth is expressed as svc-input-bandwidth and svc-output-bandwidth in bits per second. The input/output direction uses the customer site as a reference: "input bandwidth" means download bandwidth for the site, and "output bandwidth" means upload bandwidth for the site.

The service bandwidth is only configurable at the site-network-access level.

Using a different input and output bandwidth will allow the SP to determine if the customer allows for asymmetric bandwidth access, such as ADSL. It can also be used to set rate-limiting in a different way for uploading and downloading on a symmetric bandwidth access.

The bandwidth is a service bandwidth expressed primarily as IP bandwidth, but if the customer enables MPLS for Carriers' Carriers (CsC), this becomes MPLS bandwidth.

6.12.2. QoS

The model proposes to define QoS parameters in an abstracted way:

- qos-classification-policy: policy that defines a set of ordered rules to classify customer traffic.

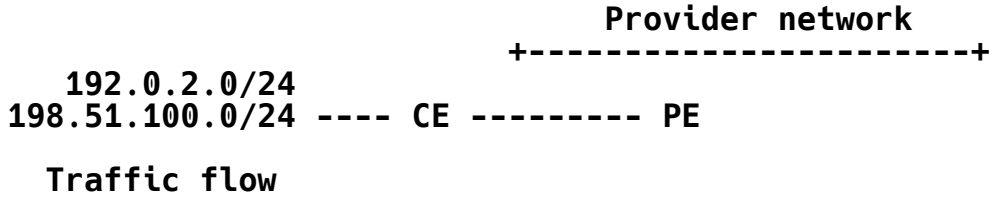
- qos-profile: QoS scheduling profile to be applied.

6.12.2.1. QoS Classification

QoS classification rules are handled by the "qos-classification-policy" container. The qos-classification-policy container is an ordered list of rules that match a flow or application and set the appropriate target class of service (target-class-id). The user can define the match using an application reference or a flow definition that is more specific (e.g., based on Layer 3 source and destination addresses, Layer 4 ports, and Layer 4 protocol). When a flow definition is used, the user can employ a "target-sites" leaf-list to identify the destination of a flow rather than using destination IP addresses. In such a case, an association between the site abstraction and the IP

addresses used by this site must be done dynamically. How this association is done is out of scope for this document; an implementation might not support this criterion and should advertise a deviation in this case. A rule that does not have a match statement is considered a match-all rule. An SP may implement a default terminal classification rule if the customer does not provide it. It will be up to the SP to determine its default target class. The current model defines some applications, but new application identities may be added through augmentation. The exact meaning of each application identity is up to the SP, so it will be necessary for the SP to advise the customer on the usage of application matching.

Where the classification is done depends on the SP's implementation of the service, but classification concerns the flow coming from the customer site and entering the network.



In the figure above, the management system should implement the classification rule:

- in the ingress direction on the PE interface, if the CE is customer-managed.
- in the ingress direction on the CE interface connected to the customer LAN, if the CE is provider-managed.

The figure below describes a sample service description of QoS classification for a site:

```
<service>
  <qos>
    <qos-classification-policy>
      <rule>
        <id>1</id>
        <match-flow>
          <ipv4-src-prefix>192.0.2.0/24</ipv4-src-prefix>
          <ipv4-dst-prefix>203.0.113.1/32</ipv4-dst-prefix>
          <l4-dst-port>80</l4-dst-port>
          <l4-protocol>tcp</l4-protocol>
        </match-flow>
        <target-class-id>DATA2</target-class-id>
      </rule>
      <rule>
        <id>2</id>
        <match-flow>
          <ipv4-src-prefix>192.0.2.0/24</ipv4-src-prefix>
          <ipv4-dst-prefix>203.0.113.1/32</ipv4-dst-prefix>
          <l4-dst-port>21</l4-dst-port>
          <l4-protocol>tcp</l4-protocol>
        </match-flow>
        <target-class-id>DATA2</target-class-id>
      </rule>
      <rule>
        <id>3</id>
        <match-application>p2p</match-application>
        <target-class-id>DATA3</target-class-id>
      </rule>
      <rule>
        <id>4</id>
        <target-class-id>DATA1</target-class-id>
      </rule>
    </qos-classification-policy>
  </qos>
</service>
```

In the example above:

- HTTP traffic from the 192.0.2.0/24 LAN destined for 203.0.113.1/32 will be classified in DATA2.

- FTP traffic from the 192.0.2.0/24 LAN destined for 203.0.113.1/32 will be classified in DATA2.

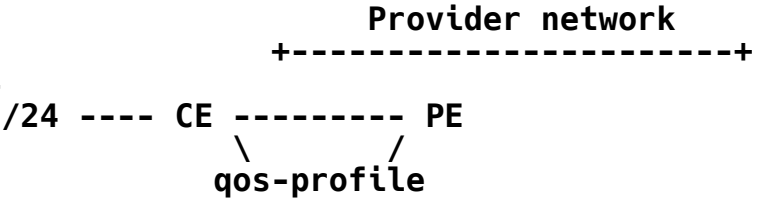
- Peer-to-peer traffic will be classified in DATA3.

- All other traffic will be classified in DATA1.

The order of rules is very important. The management system responsible for translating those rules in network element configuration MUST keep the same processing order in network element configuration. The order of rules is defined by the "id" leaf. The lowest id MUST be processed first.

6.12.2.2. QoS Profile

The user can choose either a standard profile provided by the operator or a custom profile. The "qos-profile" container defines the traffic-scheduling policy to be used by the SP.



In the case of a provider-managed or co-managed connection, the provider should ensure scheduling according to the requested policy in both traffic directions (SP to customer and customer to SP). As an example, a device-scheduling policy may be implemented on both the PE side and the CE side of the WAN link. In the case of a customer-managed connection, the provider is only responsible for ensuring scheduling from the SP network to the customer site. As an example, a device-scheduling policy may be implemented only on the PE side of the WAN link towards the customer.

A custom QoS profile is defined as a list of classes of services and associated properties. The properties are:

- rate-limit: used to rate-limit the class of service. The value is expressed as a percentage of the global service bandwidth. When the qos-profile container is implemented on the CE side, svc-output-bandwidth is taken into account as a reference. When it is implemented on the PE side, svc-input-bandwidth is used.

- latency: used to define the latency constraint of the class. The latency constraint can be expressed as the lowest possible latency or a latency boundary expressed in milliseconds. How this latency constraint will be fulfilled is up to the SP's implementation of

the service: a strict priority queuing may be used on the access and in the core network, and/or a low-latency routing configuration may be created for this traffic class.

- jitter: used to define the jitter constraint of the class. The jitter constraint can be expressed as the lowest possible jitter or a jitter boundary expressed in microseconds. How this jitter constraint will be fulfilled is up to the SP's implementation of the service: a strict priority queuing may be used on the access and in the core network, and/or a jitter-aware routing configuration may be created for this traffic class.

- bandwidth: used to define a guaranteed amount of bandwidth for the class of service. It is expressed as a percentage. The "guaranteed-bw-percent" parameter uses available bandwidth as a reference. When the qos-profile container is implemented on the CE side, svc-output-bandwidth is taken into account as a reference. When it is implemented on the PE side, svc-input-bandwidth is used. By default, the bandwidth reservation is only guaranteed at the access level. The user can use the "end-to-end" leaf to request an end-to-end bandwidth reservation, including across the MPLS transport network. (In other words, the SP will activate something in the MPLS core to ensure that the bandwidth request from the customer will be fulfilled by the MPLS core as well.) How this is done (e.g., RSVP reservation, controller reservation) is out of scope for this document.

Some constraints may not be offered by an SP; in this case, a deviation should be advertised. In addition, due to network conditions, some constraints may not be completely fulfilled by the SP; in this case, the SP should advise the customer about the limitations. How this communication is done is out of scope for this document.

Example of service configuration using a standard QoS profile:

```
<site-network-access>
 <site-network-access-id>1245HRTFGJGJ154654</site-network-access-id>
 <service>
  <svc-input-bandwidth>100000000</svc-input-bandwidth>
  <svc-output-bandwidth>100000000</svc-output-bandwidth>
  <qos>
   <qos-profile>
    <profile>PLATINUM</profile>
   </qos-profile>
  </qos>
 </service>
```

```
   </site-network-access>
   <site-network-access>
    <site-network-access-id>555555AAAA2344</site-network-access-id>
    <service>
     <svc-input-bandwidth>2000000</svc-input-bandwidth>
     <svc-output-bandwidth>2000000</svc-output-bandwidth>
     <qos>
      <qos-profile>
       <profile>GOLD</profile>
      </qos-profile>
     </qos>
    </service>
   </site-network-access>
```

Example of service configuration using a custom QoS profile:

```
   <site-network-access>
    <site-network-access-id>Site1</site-network-access-id>
    <service>
     <svc-input-bandwidth>100000000</svc-input-bandwidth>
     <svc-output-bandwidth>100000000</svc-output-bandwidth>
     <qos>
      <qos-profile>
       <classes>
        <class>
         <class-id>REAL_TIME</class-id>
         <rate-limit>10</rate-limit>
         <latency>
          <use-lowest-latency/>
         </latency>
        </class>
        <class>
         <class-id>DATA1</class-id>
         <latency>
          <latency-boundary>70</latency-boundary>
         </latency>
         <bandwidth>
          <guaranteed-bw-percent>80</guaranteed-bw-percent>
         </bandwidth>
        </class>
        <class>
         <class-id>DATA2</class-id>
         <latency>
          <latency-boundary>200</latency-boundary>
         </latency>
         <bandwidth>
          <guaranteed-bw-percent>5</guaranteed-bw-percent>
          <end-to-end/>
```

```
        </bandwidth>
       </class>
      </classes>
     </qos-profile>
    </qos>
   </service>
  </site-network-access>
```

The custom QoS profile for Site1 defines a REAL_TIME class with a latency constraint expressed as the lowest possible latency. It also defines two data classes -- DATA1 and DATA2. The two classes express a latency boundary constraint as well as a bandwidth reservation, as the REAL_TIME class is rate-limited to 10% of the service bandwidth (10% of 100 Mbps = 10 Mbps). In cases where congestion occurs, the REAL_TIME traffic can go up to 10 Mbps (let's assume that only 5 Mbps are consumed). DATA1 and DATA2 will share the remaining bandwidth (95 Mbps) according to their percentage. So, the DATA1 class will be served with at least 76 Mbps of bandwidth, while the DATA2 class will be served with at least 4.75 Mbps. The latency boundary information of the data class may help the SP define a specific buffer tuning or a specific routing within the network. The maximum percentage to be used is not limited by this model but MUST be limited by the management system according to the policies authorized by the SP.

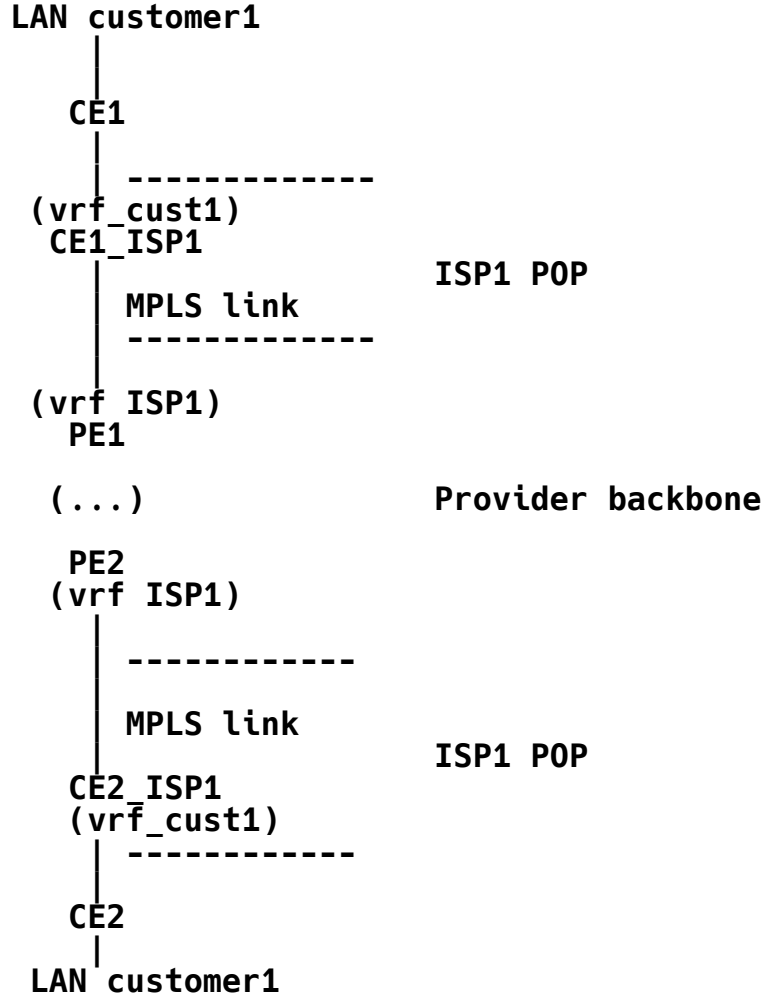
6.12.3. Multicast

The "multicast" container defines the type of site in the customer multicast service topology: source, receiver, or both. These parameters will help the management system optimize the multicast service. Users can also define the type of multicast relationship with the customer: router (requires a protocol such as PIM), host (IGMP or MLD), or both. An address family (IPv4, IPv6, or both) can also be defined.

6.13. Enhanced VPN Features

6.13.1. Carriers' Carriers

In the case of CsC [RFC4364], a customer may want to build an MPLS service using an IP VPN to carry its traffic.



In the figure above, ISP1 resells an IP VPN service but has no core network infrastructure between its POPs. ISP1 uses an IP VPN as the core network infrastructure (belonging to another provider) between its POPs.

In order to support CsC, the VPN service must indicate MPLS support by setting the "carrierscarrier" leaf to true in the vpn-service list. The link between CE1_ISP1/PE1 and CE2_ISP1/PE2 must also run an MPLS signalling protocol. This configuration is done at the site level.

In the proposed model, LDP or BGP can be used as the MPLS signalling protocol. In the case of LDP, an IGP routing protocol MUST also be activated. In the case of BGP signalling, BGP MUST also be configured as the routing protocol.

If CsC is enabled, the requested "svc-mtu" leaf will refer to the MPLS MTU and not to the IP MTU.

6.14. External ID References

The service model sometimes refers to external information through identifiers. As an example, to order a cloud-access to a particular cloud service provider (CSP), the model uses an identifier to refer to the targeted CSP. If a customer is directly using this service model as an API (through REST or NETCONF, for example) to order a particular service, the SP should provide a list of authorized identifiers. In the case of cloud-access, the SP will provide the associated identifiers for each available CSP. The same applies to other identifiers, such as std-qos-profile, OAM profile-name, and provider-profile for encryption.

How an SP provides the meanings of those identifiers to the customer is out of scope for this document.

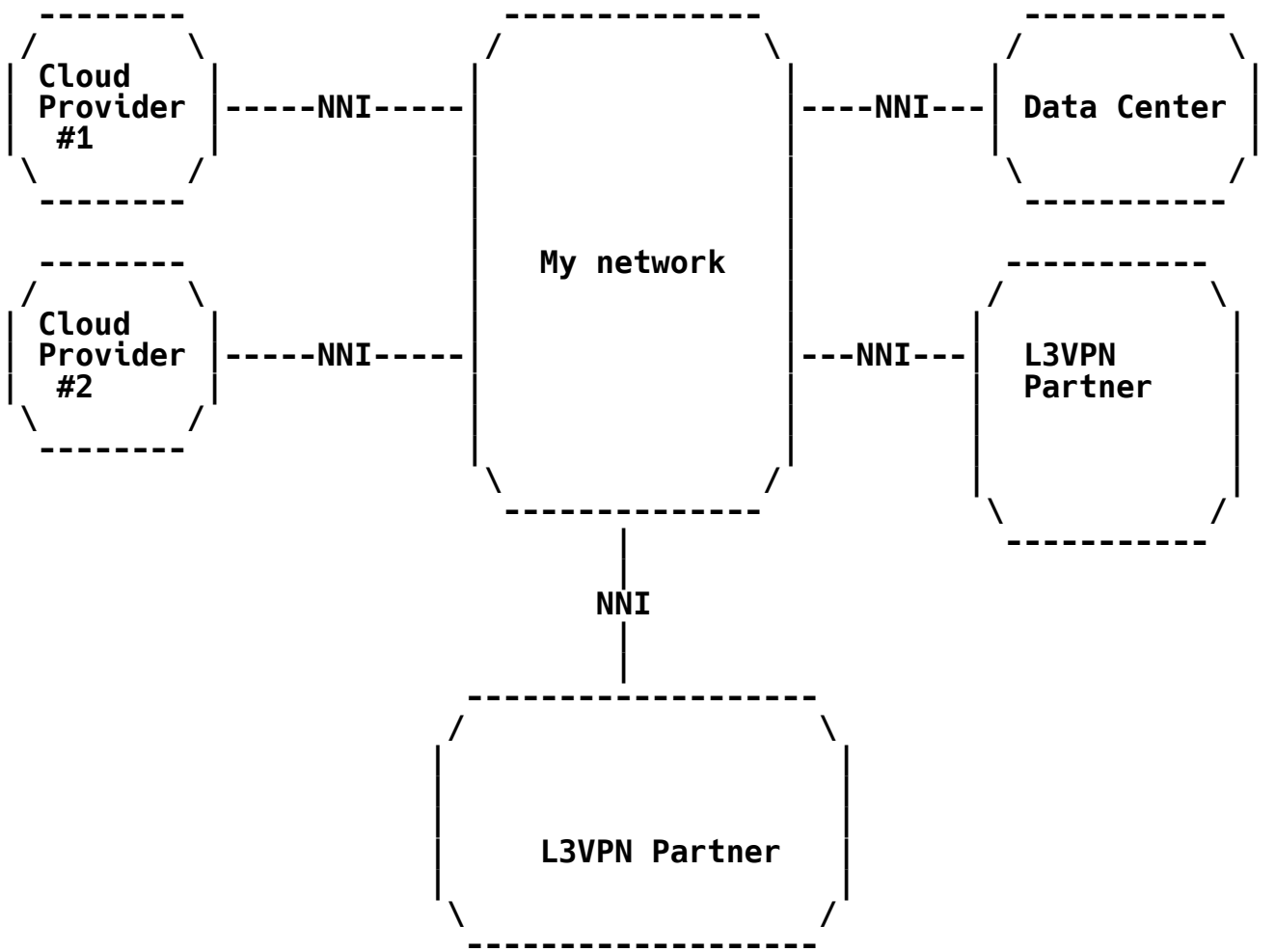
6.15. Defining NNIs

An autonomous system (AS) is a single network or group of networks that is controlled by a common system administration group and that uses a single, clearly defined routing protocol. In some cases, VPNs need to span different ASes in different geographic areas or span different SPs. The connection between ASes is established by the SPs and is seamless to the customer. Examples include

- a partnership between SPs (e.g., carrier, cloud) to extend their VPN service seamlessly.
- an internal administrative boundary within a single SP (e.g., backhaul versus core versus data center).

NNIs (network-to-network interfaces) have to be defined to extend the VPNs across multiple ASes.

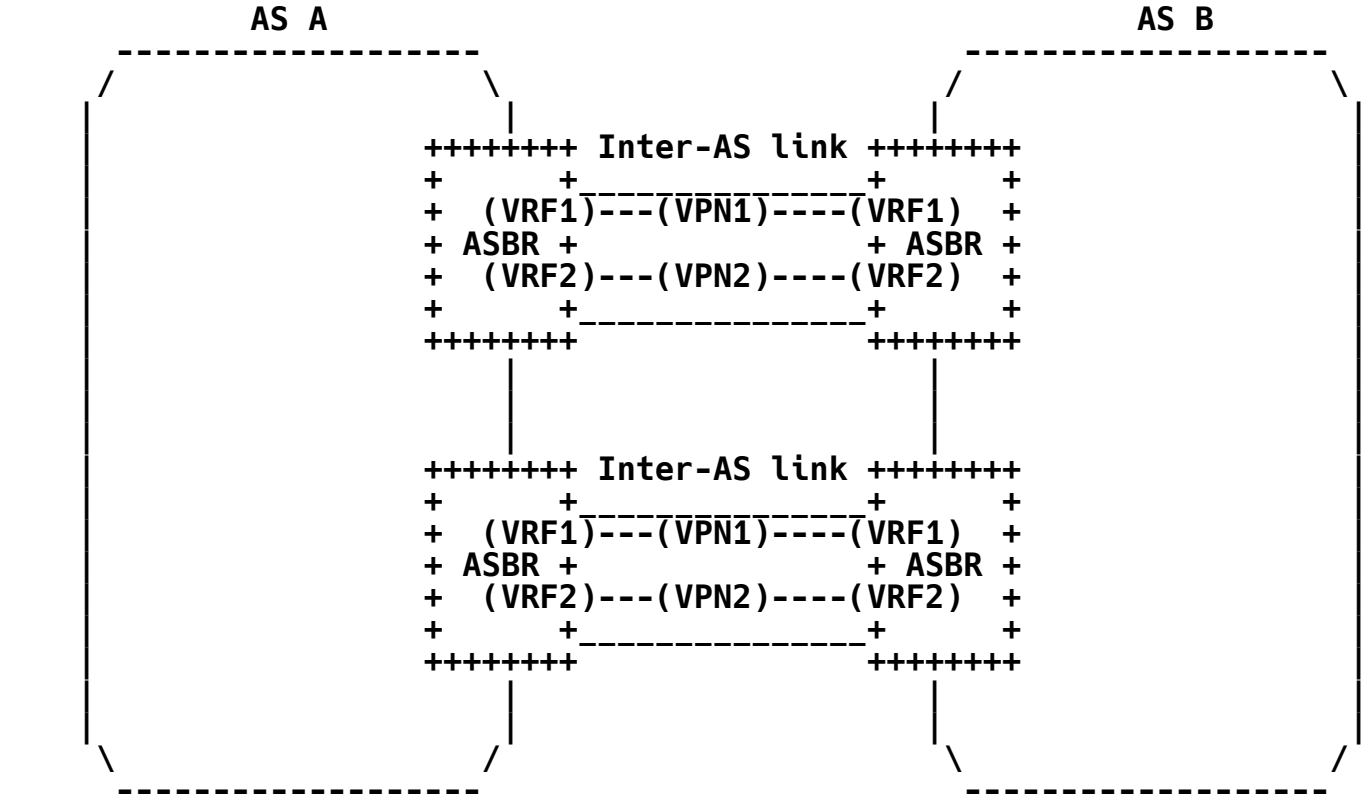
[RFC4364] defines multiple flavors of VPN NNI implementations. Each implementation has pros and cons; this topic is outside the scope of this document. For example, in an Inter-AS option A, autonomous system border router (ASBR) peers are connected by multiple interfaces with at least one of those interfaces spanning the two ASes while being present in the same VPN. In order for these ASBRs to signal unlabeled IP prefixes, they associate each interface with a VPN routing and forwarding (VRF) instance and a Border Gateway Protocol (BGP) session. As a result, traffic between the back-to-back VRFs is IP. In this scenario, the VPNs are isolated from each other, and because the traffic is IP, QoS mechanisms that operate on IP traffic can be applied to achieve customer service level agreements (SLAs).



The figure above describes an SP network called "My network" that has several NNIs. This network uses NNIs to:

- increase its footprint by relying on L3VPN partners.
- connect its own data center services to the customer IP VPN.
- enable the customer to access its private resources located in a private cloud owned by some CSPs.

6.15.1. Defining an NNI with the Option A Flavor



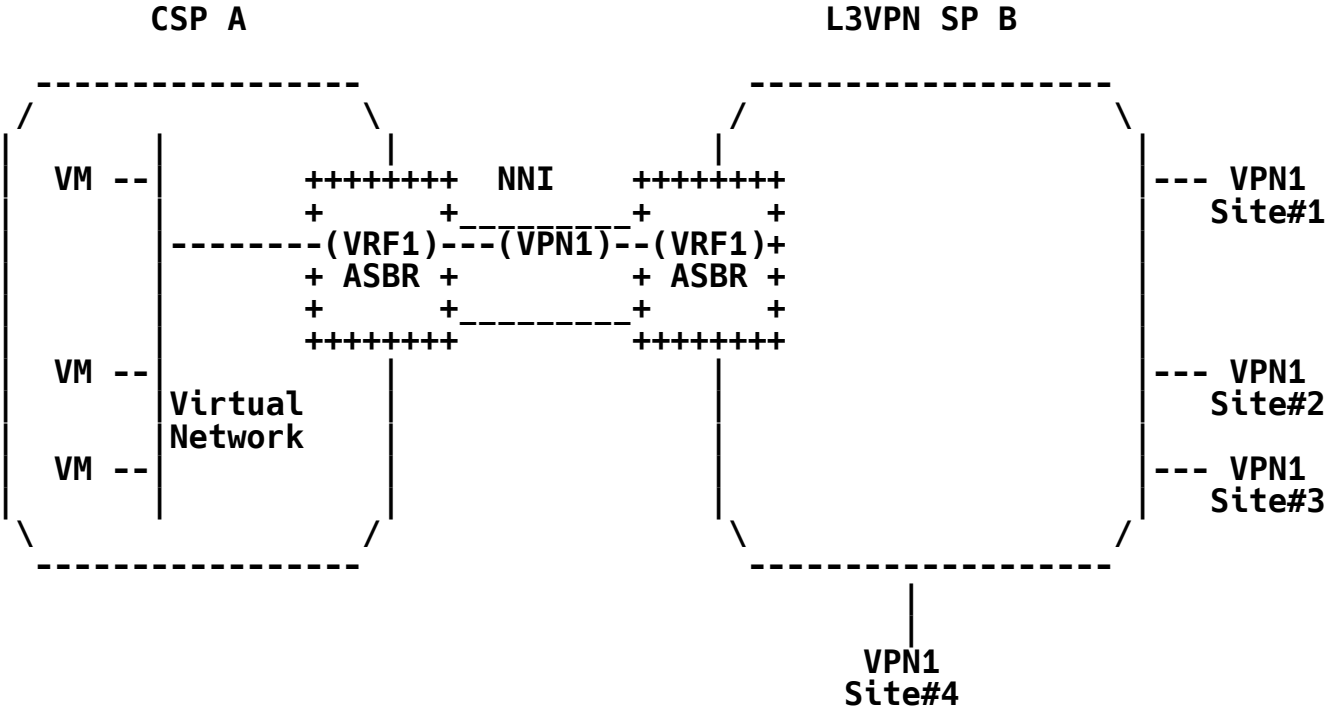
In option A, the two ASes are connected to each other with physical links on ASBRs. For resiliency purposes, there may be multiple physical connections between the ASes. A VPN connection -- physical or logical (on top of physical) -- is created for each VPN that needs to cross the AS boundary, thus providing a back-to-back VRF model.

From a service model's perspective, this VPN connection can be seen as a site. Let's say that AS B wants to extend some VPN connections for VPN C on AS A. The administrator of AS B can use this service model to order a site on AS A. All connection scenarios could be

realized using the features of the current model. As an example, the figure above shows two physical connections that have logical connections per VPN overlaid on them. This could be seen as a dual-homed subVPN scenario. Also, the administrator of AS B will be able to choose the appropriate routing protocol (e.g., E-BGP) to dynamically exchange routes between ASes.

This document assumes that the option A NNI flavor SHOULD reuse the existing VPN site modeling.

Example: a customer wants its CSP A to attach its virtual network N to an existing IP VPN (VPN1) that he has from L3VPN SP B.

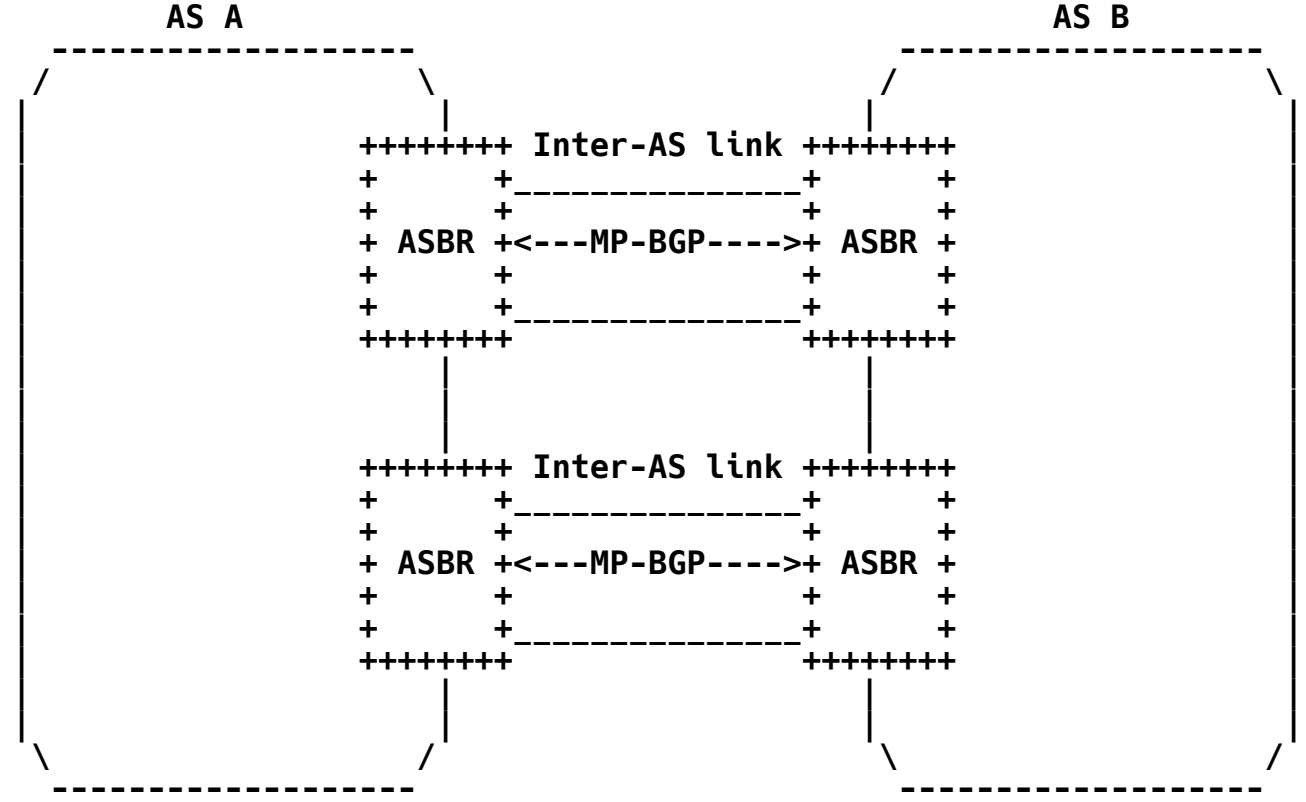


To create the VPN connectivity, the CSP or the customer may use the L3VPN service model that SP B exposes. We could consider that, as the NNI is shared, the physical connection (bearer) between CSP A and SP B already exists. CSP A may request through a service model the creation of a new site with a single site-network-access (single-homing is used in the figure). As a placement constraint, CSP A may use the existing bearer reference it has from SP A to force the placement of the VPN NNI on the existing link. The XML below illustrates a possible configuration request to SP B:

```
   <site>
       <site-id>CSP_A_attachment</site-id>
       <location>
           <city>NY</city>
           <country-code>US</country-code>
       </location>
       <site-vpn-flavor>site-vpn-flavor-nni</site-vpn-flavor>
       <routing-protocols>
         <routing-protocol>
           <type>bgp</type>
           <bgp>
               <autonomous-system>500</autonomous-system>
               <address-family>ipv4</address-family>
           </bgp>
         </routing-protocol>
       </routing-protocols>
       <site-network-accesses>
        <site-network-access>
         <site-network-access-id>CSP_A_VN1</site-network-access-id>
          <ip-connection>
           <ipv4>
            <address-allocation-type>
            static-address
            </address-allocation-type>
            <addresses>
             <provider-address>203.0.113.1</provider-address>
             <customer-address>203.0.113.2</customer-address>
             <mask>30</mask>
            </addresses>
           </ipv4>
          </ip-connection>
          <service>
           <svc-input-bandwidth>450000000</svc-input-bandwidth>
           <svc-output-bandwidth>450000000</svc-output-bandwidth>
          </service>
          <vpn-attachment>
           <vpn-id>VPN1</vpn-id>
           <site-role>any-to-any-role</site-role>
          </vpn-attachment>
        </site-network-access>
       </site-network-accesses>
       <management>
           <type>customer-managed</type>
       </management>
   </site>
```

The case described above is different from a scenario using the cloud-accesses container, as the cloud-access provides a public cloud access while this example enables access to private resources located in a CSP network.

6.15.2. Defining an NNI with the Option B Flavor



In option B, the two ASes are connected to each other with physical links on ASBRs. For resiliency purposes, there may be multiple physical connections between the ASes. The VPN "connection" between ASes is done by exchanging VPN routes through MP-BGP [RFC4760].

There are multiple flavors of implementations of such an NNI. For example:

1. The NNI is internal to the provider and is situated between a backbone and a data center. There is enough trust between the domains to not filter the VPN routes. So, all the VPN routes are exchanged. RT filtering may be implemented to save some unnecessary route states.

2. The NNI is used between providers that agreed to exchange VPN routes for specific RTs only. Each provider is authorized to use the RT values from the other provider.

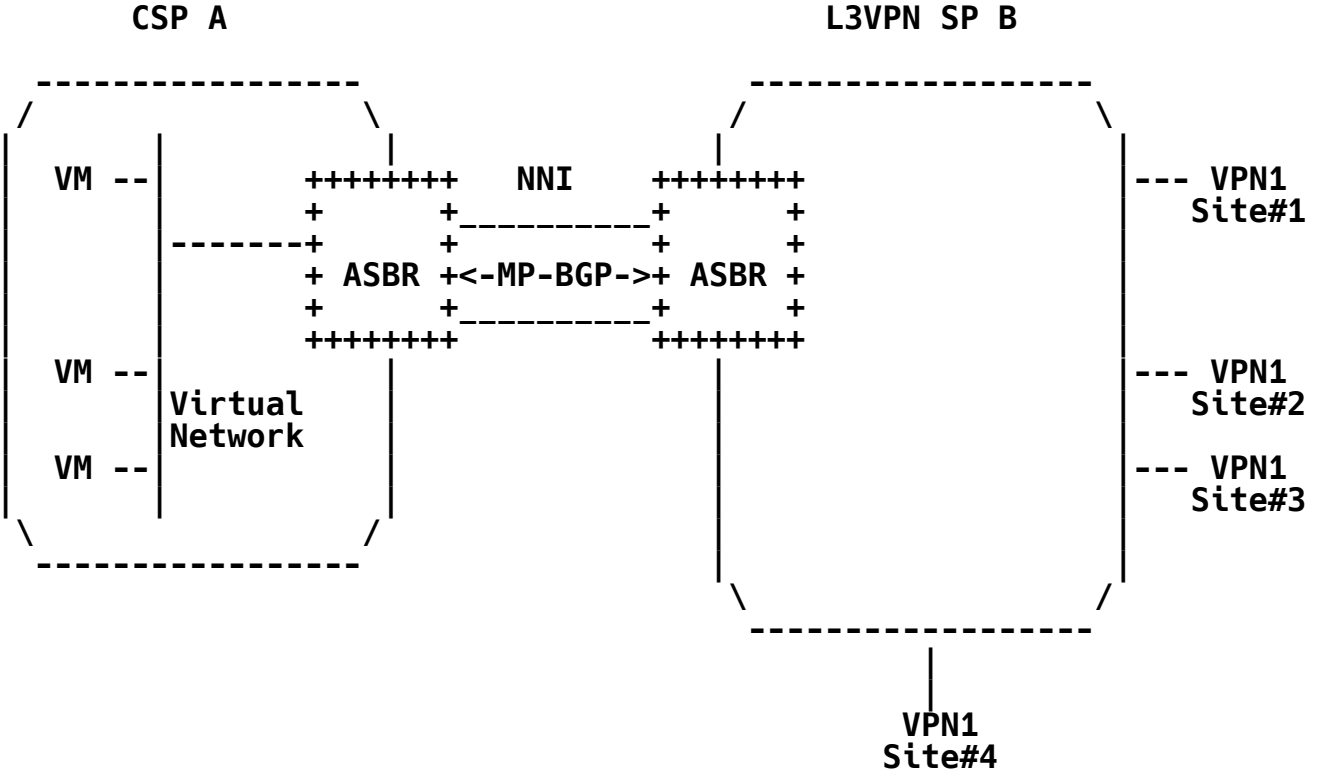
3. The NNI is used between providers that agreed to exchange VPN routes for specific RTs only. Each provider has its own RT scheme. So, a customer spanning the two networks will have different RTs in each network for a particular VPN.

Case 1 does not require any service modeling, as the protocol enables the dynamic exchange of necessary VPN routes.

Case 2 requires that an RT-filtering policy on ASBRs be maintained. From a service modeling point of view, it is necessary to agree on the list of RTs to authorize.

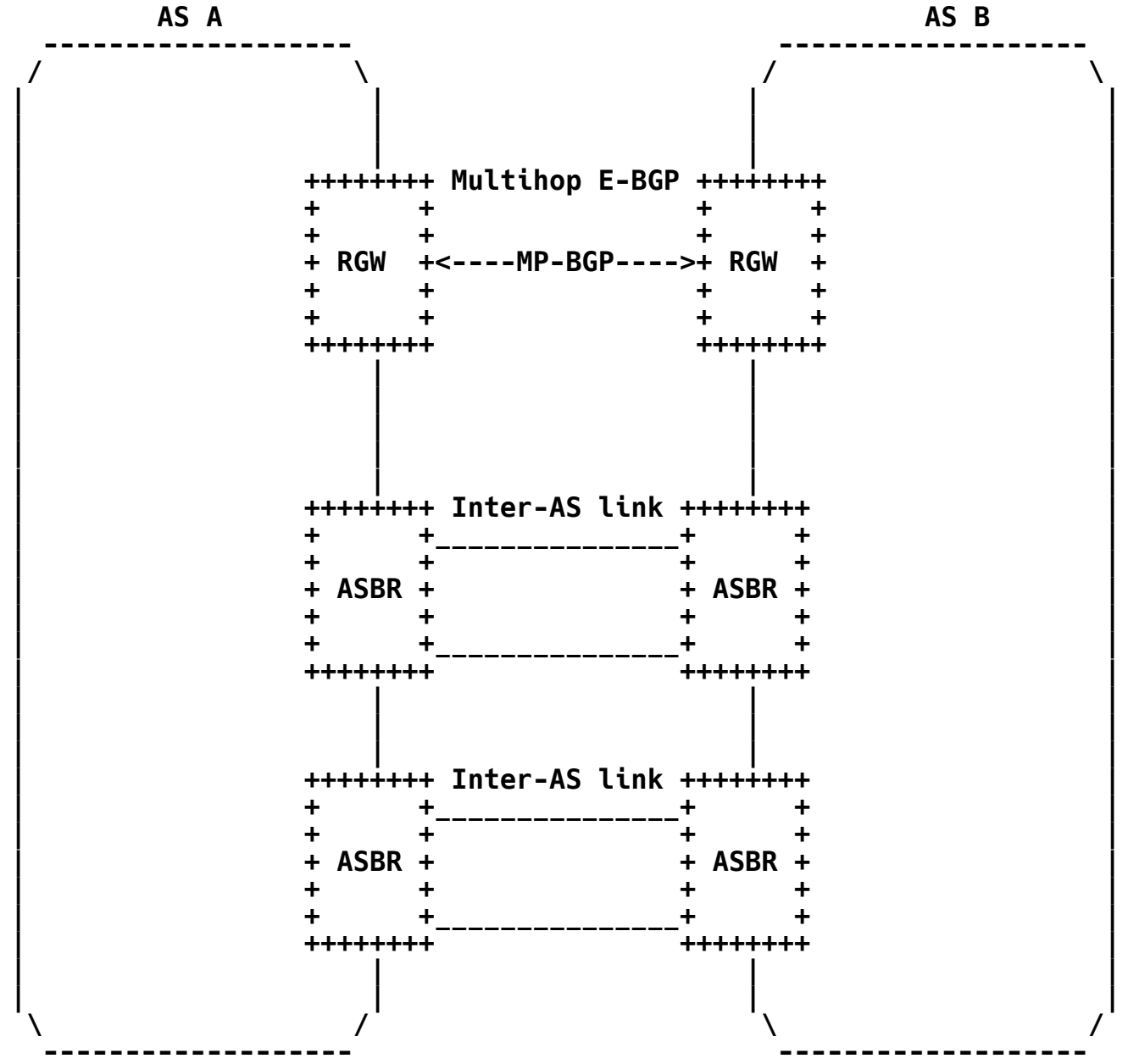
In Case 3, both ASes need to agree on the VPN RT to exchange, as well as how to map a VPN RT from AS A to the corresponding RT in AS B (and vice versa).

Those modelings are currently out of scope for this document.



The example above describes an NNI connection between CSP A and SP network B. Both SPs do not trust themselves and use a different RT allocation policy. So, in terms of implementation, the customer VPN has a different RT in each network (RT A in CSP A and RT B in SP network B). In order to connect the customer virtual network in CSP A to the customer IP VPN (VPN1) in SP network B, CSP A should request that SP network B open the customer VPN on the NNI (accept the appropriate RT). Who does the RT translation depends on the agreement between the two SPs: SP B may permit CSP A to request VPN (RT) translation.

6.15.3. Defining an NNI with the Option C Flavor



From a VPN service's perspective, the option C NNI is very similar to option B, as an MP-BGP session is used to exchange VPN routes between the ASes. The difference is that the forwarding plane and the control plane are on different nodes, so the MP-BGP session is multihop between routing gateway (RGW) nodes.

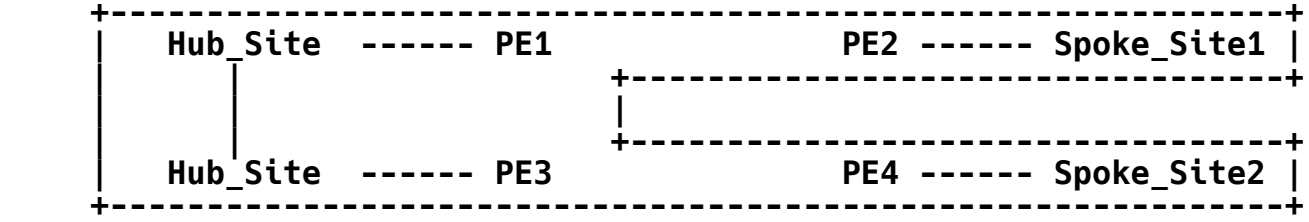
From a VPN service's point of view, modeling options B and C will be identical.

7. Service Model Usage Example

As explained in Section 5, this service model is intended to be instantiated at a management layer and is not intended to be used directly on network elements. The management system serves as a central point of configuration of the overall service.

This section provides an example 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 VPN service for three sites using a Hub-and-Spoke VPN service topology. One of the sites will be dual-homed, and load-sharing is expected.



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

```
<vpn-service>
    <vpn-id>12456487</vpn-id>
    <vpn-service-topology>hub-spoke</vpn-service-topology>
</vpn-service>
```

When receiving the request for provisioning the VPN service, the management system will internally (or through communication with another OSS component) allocate VPN RTs. In this specific case, two RTs will be allocated (100:1 for Hub and 100:2 for Spoke). The output below describes the configuration of Spoke_Site1.

```
<site>
    <site-id>Spoke_Site1</site-id>
    <location>
        <city>NY</city>
        <country-code>US</country-code>
    </location>
    <routing-protocols>
      <routing-protocol>
        <type>bgp</type>
        <bgp>
            <autonomous-system>500</autonomous-system>
            <address-family>ipv4</address-family>
            <address-family>ipv6</address-family>
        </bgp>
      </routing-protocol>
    </routing-protocols>
    <site-network-accesses>
     <site-network-access>
      <site-network-access-id>Spoke_Site1</site-network-access-id>
      <access-diversity>
       <groups>
        <group>
         <group-id>20</group-id>
        </group>
       </groups>
       <constraints>
        <constraint>
         <constraint-type>pe-diverse</constraint-type>
         <target>
          <group>
           <group-id>10</group-id>
          </group>
         </target>
        </constraint>
       </constraints>
      </access-diversity>
      <ip-connection>
        <ipv4>
         <address-allocation-type>
         static-address
         </address-allocation-type>
```

```
            <addresses>
             <provider-address>203.0.113.254</provider-address>
             <customer-address>203.0.113.2</customer-address>
             <mask>24</mask>
            </addresses>
           </ipv4>
           <ipv6>
            <address-allocation-type>
            static-address
            </address-allocation-type>
             <addresses>
              <provider-address>2001:db8::1</provider-address>
              <customer-address>2001:db8::2</customer-address>
              <mask>64</mask>
             </addresses>
           </ipv6>
         </ip-connection>
         <service>
           <svc-input-bandwidth>450000000</svc-input-bandwidth>
           <svc-output-bandwidth>450000000</svc-output-bandwidth>
         </service>
         <vpn-attachment>
           <vpn-id>12456487</vpn-id>
           <site-role>spoke-role</site-role>
         </vpn-attachment>
       </site-network-access>
     </site-network-accesses>
     <management>
         <type>provider-managed</type>
     </management>
   </site>
```

When receiving the request for provisioning Spoke_Site1, the management system MUST allocate network resources for this site. It MUST first determine the target network elements to provision the access, particularly the PE router (and perhaps also an aggregation switch). As described in Section 6.6, the management system SHOULD use the location information and SHOULD use the access-diversity constraint to find the appropriate PE. In this case, we consider that Spoke_Site1 requires PE diversity with the Hub and that the management system allocates PEs based on the least distance. Based on the location information, the management system finds the available PEs in the area nearest the customer and picks one that fits the access-diversity constraint.

When the PE is chosen, the management system needs to allocate interface resources on the node. One interface is selected from the pool of available PEs. The management system can start provisioning the chosen PE node via whatever means the management system prefers (e.g., NETCONF, CLI). The management system will check to see if a VRF that fits its needs is already present. If not, it will provision the VRF: the RD will be derived from the internal allocation policy model, and the RTs will be derived from the VPN policy configuration of the site (the management system allocated some RTs for the VPN). As the site is a Spoke site (site-role), the management system knows which RTs must be imported and exported. As the site is provider-managed, some management RTs may also be added (100:5000). Standard provider VPN policies MAY also be added in the configuration.

Example of generated PE configuration:

```
ip vrf Customer1
 export-map STD-CUSTOMER-EXPORT      <---- Standard SP configuration
 route-distinguisher 100:3123234324
 route-target import 100:1
 route-target import 100:5000        <---- Standard SP configuration
 route-target export 100:2                   for provider-managed CE
!
```

When the VRF has been provisioned, the management system can start configuring the access on the PE using the allocated interface information. IP addressing is chosen by the management system. One address will be picked from an allocated subnet for the PE, and another will be used for the CE configuration. Routing protocols will also be configured between the PE and CE; because this model is provider-managed, the choices are left to the SP. BGP was chosen for this example. This choice is independent of the routing protocol chosen by the customer. BGP will be used to configure the CE-to-LAN connection as requested in the service model. Peering addresses will be derived from those of the connection. As the CE is provider-managed, the CE's AS number can be automatically allocated by the management system. Standard configuration templates provided by the SP may also be added.

Example of generated PE configuration:

```
interface Ethernet1/1/0.10
 encapsulation dot1q 10
 ip vrf forwarding Customer1
 ip address 198.51.100.1 255.255.255.252 <---- Comes from
                                              automated allocation
 ipv6 address 2001:db8::10:1/64
 ip access-group STD-PROTECT-IN     <---- Standard SP config
!
router bgp 100
 address-family ipv4 vrf Customer1
  neighbor 198.51.100.2 remote-as 65000   <---- Comes from
                                             automated allocation
  neighbor 198.51.100.2 route-map STD in  <---- Standard SP config
  neighbor 198.51.100.2 filter-list 10 in <---- Standard SP config
!
 address-family ipv6 vrf Customer1
  neighbor 2001:db8::0a10:2 remote-as 65000   <---- Comes from
                                                automated allocation
  neighbor 2001:db8::0a10:2 route-map STD in  <---- Standard SP
                                                    config
  neighbor 2001:db8::0a10:2 filter-list 10 in <---- Standard SP
                                                    config
!
ip route vrf Customer1 192.0.2.1 255.255.255.255 198.51.100.2
! Static route for provider administration of CE
!
```

As the CE router is not reachable at this stage, the management system can produce a complete CE configuration that can be manually uploaded to the node before sending the CE configuration to the customer premises. The CE configuration will be built in the same way as the PE would be configured. Based on the CE type (vendor/model) allocated to the customer as well as the bearer information, the management system knows which interface must be configured on the CE. PE-CE link configuration is expected to be handled automatically using the SP OSS, as both resources are managed internally. CE-to-LAN-interface parameters such as IP addressing are derived from the ip-connection container, taking into account how the management system distributes addresses between the PE and CE within the subnet. This will allow a plug-and-play configuration for the CE to be created.

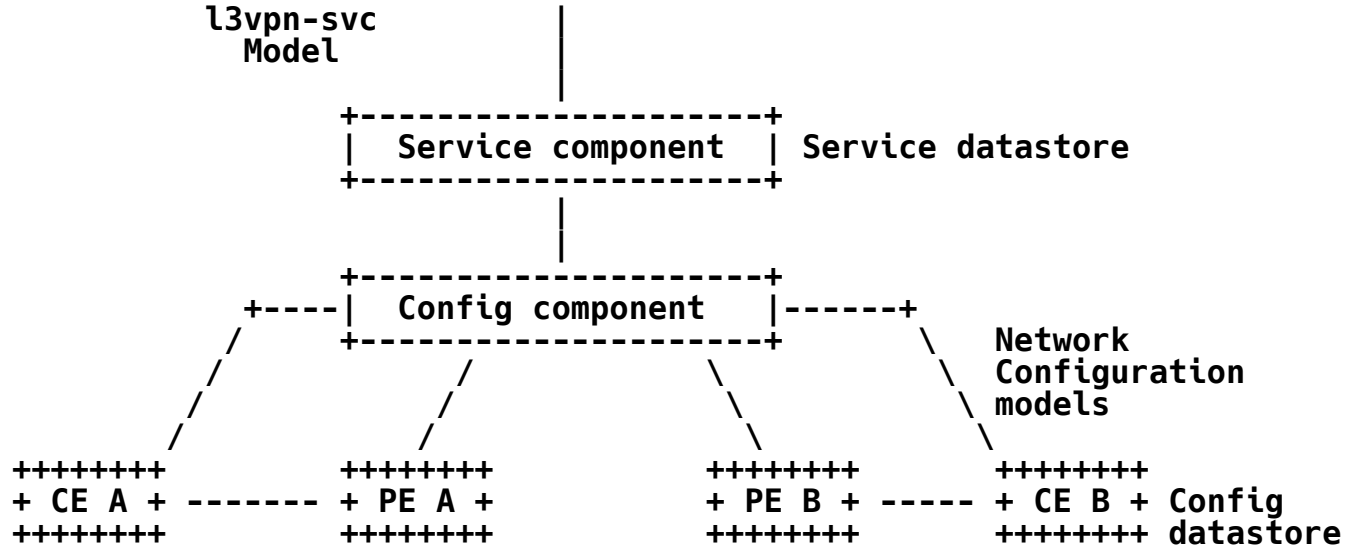
Example of generated CE configuration:

```
interface Loopback10
 description "Administration"
 ip address 192.0.2.1 255.255.255.255
!
interface FastEthernet10
 description "WAN"
 ip address 198.51.100.2 255.255.255.252 <---- Comes from
                                              automated allocation
 ipv6 address 2001:db8::0a10:2/64
!
interface FastEthernet11
 description "LAN"
 ip address 203.0.113.254 255.255.255.0 <---- Comes from the
                                           ip-connection container
 ipv6 address 2001:db8::1/64
!
router bgp 65000
 address-family ipv4
  redistribute static route-map STATIC2BGP <---- Standard SP
                                                 configuration
  neighbor 198.51.100.1 remote-as 100     <---- Comes from
                                           automated allocation
  neighbor 203.0.113.2 remote-as 500     <---- Comes from the
                                           ip-connection container
 address-family ipv6
  redistribute static route-map STATIC2BGP <---- Standard SP
                                                 configuration
  neighbor 2001:db8::0a10:1 remote-as 100     <---- Comes from
                                           automated allocation
  neighbor 2001:db8::2 remote-as 500     <---- Comes from the
                                           ip-connection container
!
route-map STATIC2BGP permit 10
 match tag 10
!
```

8. Interaction with Other YANG Modules

As expressed in Section 5, this 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, so the component instantiating the service model (let's call it "service component") and the component responsible for network element configuration (let's call it "configuration component") may be different.



In the previous sections, we provided some examples of the translation of service provisioning requests to router configuration lines. In the NETCONF/YANG ecosystem, we expect NETCONF/YANG to be used between the configuration component and network elements to configure the requested services on those elements.

In this framework, specifications are expected to provide specific YANG modeling of service components on network elements. There will be a strong relationship between the abstracted view provided by this service model and the detailed configuration view that will be provided by specific configuration models for network elements.

The authors of this document anticipate definitions of YANG models for the network elements listed below. Note that this list is not exhaustive:

- VRF definition, including VPN policy expression.
- Physical interface.
- IP layer (IPv4, IPv6).
- QoS: classification, profiles, etc.
- Routing protocols: support of configuration of all protocols listed in the document, as well as routing policies associated with those protocols.
- Multicast VPN.
- Network address translation.

Example of a VPN site request at the service level, using this model:

```
<site>
 <site-id>Site A</site-id>
 <site-network-accesses>
  <site-network-access>
   <ip-connection>
    <ipv4>
     <address-allocation-type>
     static-address
     </address-allocation-type>
     <addresses>
      <provider-address>203.0.113.254</provider-address>
      <customer-address>203.0.113.2</customer-address>
      <mask>24</mask>
     </addresses>
    </ipv4>
   </ip-connection>
   <vpn-attachment>
    <vpn-policy-id>VPNPOL1</vpn-policy-id>
   </vpn-attachment>
  </site-network-access>
 </site-network-accesses>
```

```
    <routing-protocols>
     <routing-protocol>
      <type>static</type>
      <static>
       <cascaded-lan-prefixes>
        <ipv4-lan-prefixes>
         <lan>198.51.100.0/30</lan>
         <next-hop>203.0.113.2</next-hop>
        </ipv4-lan-prefixes>
       </cascaded-lan-prefixes>
      </static>
     </routing-protocol>
    </routing-protocols>
    <management>
     <type>customer-managed</type>
    </management>
    <vpn-policies>
     <vpn-policy>
      <vpn-policy-id>VPNPOL1</vpn-policy-id>
      <entries>
       <id>1</id>
        <vpn>
         <vpn-id>VPN1</vpn-id>
         <site-role>any-to-any-role</site-role>
        </vpn>
       </entries>
     </vpn-policy>
    </vpn-policies>
   </site>
```

In the service example above, the service component is expected to request that the configuration component of the management system provide the configuration of the service elements. If we consider that the service component selected a PE (PE A) as the target PE for the site, the configuration component will need to push the configuration to PE A. The configuration component will use several YANG data models to define the configuration to be applied to PE A. The XML configuration of PE A might look like this:

```
   <if:interfaces>
        <if:interface>
         <if:name>eth0</if:name>
         <if:type>ianaift:ethernetCsmacd</if:type>
         <if:description>
          Link to CE A.
         </if:description>
         <ip:ipv4>
          <ip:address>
```

```
          <ip:ip>203.0.113.254</ip:ip>
          <ip:prefix-length>24</ip:prefix-length>
         </ip:address>
         <ip:forwarding>true</ip:forwarding>
        </ip:ipv4>
       </if:interface>
   </if:interfaces>
   <rt:routing>
       <rt:routing-instance>
         <rt:name>VRF_CustA</rt:name>
         <rt:type>l3vpn-network:vrf</rt:type>
         <rt:description>VRF for Customer A</rt:description>
         <l3vpn-network:route-distinguisher>
         100:1546542343
         </l3vpn-network:route-distinguisher>
         <l3vpn-network:import-rt>100:1</l3vpn-network:import-rt>
         <l3vpn-network:export-rt>100:1</l3vpn-network:export-rt>
         <rt:interfaces>
          <rt:interface>
           <rt:name>eth0</rt:name>
          </rt:interface>
         </rt:interfaces>
         <rt:routing-protocols>
          <rt:routing-protocol>
           <rt:type>rt:static</rt:type>
           <rt:name>st0</rt:name>
           <rt:static-routes>
            <v4ur:ipv4>
             <v4ur:route>
              <v4ur:destination-prefix>
              198.51.100.0/30
              </v4ur:destination-prefix>
              <v4ur:next-hop>
               <v4ur:next-hop-address>
               203.0.113.2
               </v4ur:next-hop-address>
              </v4ur:next-hop>
             </v4ur:route>
            </v4ur:ipv4>
           </rt:static-routes>
          </rt:routing-protocol>
         </rt:routing-protocols>
       </rt:routing-instance>
   </rt:routing>
```

9. YANG Module

```
<CODE BEGINS>

file "ietf-l3vpn-svc@2017-01-27.yang"

module ietf-l3vpn-svc {
 namespace "urn:ietf:params:xml:ns:yang:ietf-l3vpn-svc";

 prefix l3vpn-svc;

 import ietf-inet-types {
  prefix inet;
 }

 import ietf-yang-types {
  prefix yang;
 }

 organization
  "IETF L3SM Working Group";

 contact
  "WG List: <mailto:l3sm@ietf.org>

  Editor:
   L3SM WG

  Chairs:
   Adrian Farrel, Qin Wu
  ";

 description
  "This YANG module defines a generic service configuration
  model for Layer 3 VPNs.  This model is common across all
  vendor implementations.";

 revision 2017-01-27 {
  description
   "Initial document.";
  reference
    "RFC 8049.";
 }
```

```
    /* Features */

    feature cloud-access {
     description
      "Allows the VPN to connect to a CSP.";
    }
    feature multicast {
     description
      "Enables multicast capabilities in a VPN.";
    }
    feature ipv4 {
     description
      "Enables IPv4 support in a VPN.";
    }
    feature ipv6 {
     description
      "Enables IPv6 support in a VPN.";
    }
    feature carrierscarrier {
     description
      "Enables support of CsC.";
    }
    feature extranet-vpn {
     description
      "Enables support of extranet VPNs.";
    }
    feature site-diversity {
     description
      "Enables support of site diversity constraints.";
    }
    feature encryption {
     description
      "Enables support of encryption.";
    }
    feature qos {
     description
      "Enables support of classes of services.";
    }
    feature qos-custom {
     description
      "Enables support of the custom QoS profile.";
    }
    feature rtg-bgp {
     description
      "Enables support of the BGP routing protocol.";
    }
```

```
    feature rtg-rip {
     description
      "Enables support of the RIP routing protocol.";
    }
    feature rtg-ospf {
     description
      "Enables support of the OSPF routing protocol.";
    }
    feature rtg-ospf-sham-link {
     description
      "Enables support of OSPF sham links.";
    }
    feature rtg-vrrp {
     description
      "Enables support of the VRRP routing protocol.";
    }
    feature fast-reroute {
     description
      "Enables support of Fast Reroute.";
    }
    feature bfd {
     description
      "Enables support of BFD.";
    }
    feature always-on {
     description
      "Enables support of the 'always-on' access constraint.";
    }
    feature requested-type {
     description
      "Enables support of the 'requested-type' access constraint.";
    }
    feature bearer-reference {
     description
      "Enables support of the 'bearer-reference' access constraint.";
    }

    /* Typedefs */

    typedef svc-id {
     type string;
     description
      "Defines a type of service component identifier.";
    }
```

```
    typedef template-id {
     type string;
     description
      "Defines a type of service template 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.";
    }

    /* Identities */

    identity site-network-access-type {
     description
      "Base identity for site-network-access type.";
    }
    identity point-to-point {
     base site-network-access-type;
     description
      "Identity for point-to-point connection.";
    }
    identity multipoint {
     base site-network-access-type;
     description
      "Identity for multipoint connection.
      Example: Ethernet broadcast segment.";
    }
    identity placement-diversity {
     description
      "Base identity for site placement constraints.";
    }
    identity bearer-diverse {
     base placement-diversity;
     description
      "Identity for bearer diversity.
      The bearers should not use common elements.";
    }
```

```
    identity pe-diverse {
     base placement-diversity;
     description
      "Identity for PE diversity.";
    }
    identity pop-diverse {
     base placement-diversity;
     description
      "Identity for POP diversity.";
    }
    identity linecard-diverse {
     base placement-diversity;
     description
      "Identity for linecard diversity.";
    }
    identity same-pe {
     base placement-diversity;
     description
      "Identity for having sites connected on the same PE.";
    }
    identity same-bearer {
     base placement-diversity;
     description
      "Identity for having sites connected using the same bearer.";
    }
    identity 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 site-vpn-flavor {
     description
      "Base identity for the site VPN service flavor.";
    }
    identity site-vpn-flavor-single {
     base site-vpn-flavor;
     description
      "Base identity for the site VPN service flavor.
      Used when the site belongs to only one VPN.";
    }
```

```
    identity site-vpn-flavor-multi {
     base site-vpn-flavor;
     description
      "Base identity for the site VPN service flavor.
       Used when a logical connection of a site
       belongs to multiple VPNs.";
    }
    identity site-vpn-flavor-sub {
     base site-vpn-flavor;
     description
      "Base identity for the site VPN service flavor.
       Used when a site has multiple logical connections.
       Each connection may belong to different multiple VPNs.";
    }
    identity site-vpn-flavor-nni {
     base site-vpn-flavor;
     description
      "Base identity for the site VPN service flavor.
       Used to describe an NNI option A connection.";
    }
    identity management {
     description
      "Base identity for site management scheme.";
    }
    identity co-managed {
     base management;
     description
      "Base identity for co-managed site.";
    }
    identity customer-managed {
     base management;
     description
      "Base identity for customer-managed site.";
    }
    identity provider-managed {
     base management;
     description
      "Base identity for provider-managed site.";
    }
    identity address-allocation-type {
     description
      "Base identity for address-allocation-type for PE-CE link.";
    }
    identity provider-dhcp {
     base address-allocation-type;
     description
      "Provider network provides DHCP service to customer.";
    }
```

```
    identity provider-dhcp-relay {
     base address-allocation-type;
     description
      "Provider network provides DHCP relay service to customer.";
    }
    identity provider-dhcp-slaac {
     base address-allocation-type;
     description
      "Provider network provides DHCP service to customer,
      as well as SLAAC.";
    }
    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 site-role {
     description
      "Base identity for site type.";
    }
    identity any-to-any-role {
     base site-role;
     description
      "Site in an any-to-any IP VPN.";
    }
    identity spoke-role {
     base site-role;
     description
      "Spoke site in a Hub-and-Spoke IP VPN.";
    }
    identity hub-role {
     base site-role;
     description
      "Hub site in a Hub-and-Spoke IP VPN.";
    }
```

```
    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 hub-spoke-disjoint {
     base vpn-topology;
     description
      "Identity for Hub-and-Spoke VPN topology
       where Hubs cannot communicate with each other.";
    }

    identity multicast-tree-type {
     description
      "Base identity for multicast tree type.";
    }
    identity ssm-tree-type {
     base multicast-tree-type;
     description
      "Identity for SSM tree type.";
    }
    identity asm-tree-type {
     base multicast-tree-type;
     description
      "Identity for ASM tree type.";
    }
    identity bidir-tree-type {
     base multicast-tree-type;
     description
      "Identity for bidirectional tree type.";
    }

    identity multicast-rp-discovery-type {
     description
      "Base identity for RP discovery type.";
    }
```

```
    identity auto-rp {
     base multicast-rp-discovery-type;
     description
      "Base identity for Auto-RP discovery type.";
    }
    identity static-rp {
     base multicast-rp-discovery-type;
     description
      "Base identity for static type.";
    }
    identity bsr-rp {
     base multicast-rp-discovery-type;
     description
      "Base identity for BSR discovery type.";
    }

    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 rip {
     base routing-protocol-type;
     description
      "Identity for RIP protocol type.";
    }
    identity vrrp {
     base routing-protocol-type;
     description
      "Identity for VRRP protocol type.
      This is to be used when LANs are directly connected
      to PE routers.";
    }
```

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

    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.";
    }

    /* Groupings */

    grouping vpn-service-cloud-access {
     container cloud-accesses {
     if-feature cloud-access;
     list cloud-access {

      key cloud-identifier;

      leaf cloud-identifier {
       type string;
       description
        "Identification of cloud service.
        Local administration meaning.";
      }
      choice list-flavor {
       case permit-any {
        leaf permit-any {
         type empty;
         description
          "Allows all sites.";
        }
       }
       case deny-any-except {
        leaf-list permit-site {
         type leafref {
          path "/l3vpn-svc/sites/site/site-id";
         }
         description
          "Site ID to be authorized.";
        }
       }
```

```
        case permit-any-except {
         leaf-list deny-site {
          type leafref {
           path "/l3vpn-svc/sites/site/site-id";
          }
          description
           "Site ID to be denied.";
         }
        }
        description
         "Choice for cloud access policy.";
       }
       container authorized-sites {
        list authorized-site {
         key site-id;

         leaf site-id {
          type leafref {
           path "/l3vpn-svc/sites/site/site-id";
          }
          description
           "Site ID.";
         }
         description
          "List of authorized sites.";
        }
        description
         "Configuration of authorized sites.";
       }
       container denied-sites {
        list denied-site {
         key site-id;

         leaf site-id {
          type leafref {
           path "/l3vpn-svc/sites/site/site-id";
          }
          description
           "Site ID.";
         }
         description
          "List of denied sites.";
        }
        description
         "Configuration of denied sites.";
       }
```

```
      container address-translation {
       container nat44 {
        leaf enabled {
         type boolean;
         default false;
         description
          "Controls whether or not address translation is required.";
        }
        leaf nat44-customer-address {
         type inet:ipv4-address;
         must "../enabled = 'true'" {
          description
           "Applicable only if address translation is enabled.";
         }
         description
          "Address to be used for translation.
          This is to be used if the customer is
          providing the address.";
        }
        description
         "IPv4-to-IPv4 translation.";
       }
       description
        "Container for NAT.";
      }
      description
       "Cloud access configuration.";
     }
      description
       "Container for cloud access configurations.";
     }
     description
      "Grouping for VPN cloud definition.";
    }
```

```
    grouping multicast-rp-group-cfg {
     choice group-format {
      case startend {
       leaf group-start {
        type inet:ip-address;
        description
         "First group address.";
       }
       leaf group-end {
        type inet:ip-address;
        description
         "Last group address.";
       }
      }
      case singleaddress {
       leaf group-address {
        type inet:ip-address;
        description
         "Group address.";
       }
      }
      description
       "Choice for group format.";
     }
     description
      "Definition of groups for RP-to-group mapping.";
    }

    grouping vpn-service-multicast {
     container multicast {
      if-feature multicast;
      leaf enabled {
       type boolean;
       default false;
       description
        "Enables multicast.";
      }
      container customer-tree-flavors {
       leaf-list tree-flavor {
         type identityref {
          base multicast-tree-type;
         }
         description
          "Type of tree to be used.";
       }
       description
        "Type of trees used by customer.";
      }
```

```
      container rp {
       container rp-group-mappings {
        list rp-group-mapping {
         key id;

         leaf id {
          type uint16;
          description
           "Unique identifier for the mapping.";
         }
         container provider-managed {
          leaf enabled {
           type boolean;
           default false;
           description
            "Set to true if the RP must be a provider-managed node.
             Set to false if it is a customer-managed node.";
          }
          leaf rp-redundancy {
           when "../enabled = 'true'" {
            description
             "Relevant when the RP is provider-managed.";
           }
           type boolean;
           default false;
           description
            "If true, a redundancy mechanism for the RP is required.";
          }
          leaf optimal-traffic-delivery {
           when "../enabled = 'true'" {
            description
             "Relevant when the RP is provider-managed.";
           }
           type boolean;
           default false;
           description
            "If true, the SP must ensure that
             traffic uses an optimal path.";
          }
          description
           "Parameters for a provider-managed RP.";
         }
```

```
          leaf rp-address {
           when "../provider-managed/enabled = 'false'" {
            description
             "Relevant when the RP is provider-managed.";
           }
           type inet:ip-address;
           description
            "Defines the address of the RP.
            Used if the RP is customer-managed.";
          }

          container groups {
           list group {
            key id;

            leaf id {
             type uint16;
             description
              "Identifier for the group.";
            }
            uses multicast-rp-group-cfg;
            description
             "List of groups.";
           }

           description
            "Multicast groups associated with the RP.";
          }

          description
           "List of RP-to-group mappings.";
         }
         description
          "RP-to-group mappings.";
        }
        container rp-discovery {
         leaf rp-discovery-type {
          type identityref {
           base multicast-rp-discovery-type;
          }
          default static-rp;
          description
           "Type of RP discovery used.";
         }
```

```
        container bsr-candidates {
         when "../rp-discovery-type = 'bsr-rp'" {
          description
           "Only applicable if discovery type is BSR-RP.";
         }
         leaf-list bsr-candidate-address {
          type inet:ip-address;
          description
           "Address of BSR candidate.";
         }
         description
          "Customer BSR candidate's address.";
        }
        description
         "RP discovery parameters.";
       }

       description
        "RP parameters.";
      }
      description
       "Multicast global parameters for the VPN service.";
     }
     description
      "Grouping for multicast VPN definition.";
    }

    grouping vpn-service-mpls {
     leaf carrierscarrier {
      if-feature carrierscarrier;
      type boolean;
      default false;
      description
       "The VPN is using CsC, and so MPLS is required.";
     }
     description
      "Grouping for MPLS CsC definition.";
    }
```

```
    grouping customer-location-info {
     container locations {
      list location {
       key location-id;

       leaf location-id {
        type svc-id;
        description
         "Identifier for a particular location.";
       }
       leaf address {
        type string;
        description
         "Address (number and street) of the site.";
       }
       leaf postal-code {
        type string;
        description
         "Postal code of the site.";
       }
       leaf state {
        type string;
        description
         "State of the site.  This leaf can also be used to describe
          a region for a country that does not have states.";
       }
       leaf city {
        type string;
        description
         "City of the site.";
       }
       leaf country-code {
        type string {
         pattern '[A-Z]{2}';
        }
        description
         "Country of the site.
          Expressed as ISO ALPHA-2 code.";
       }
       description
        "Location of the site.";
       }
       description
        "List of locations for the site.";
     }
     description
      "This grouping defines customer location parameters.";
    }
```

```
    grouping site-group {
     container groups {
      list group {
       key group-id;

       leaf group-id {
        type string;
        description
         "Group-id the site belongs to.";
       }
       description
        "List of group-ids.";
      }
      description
       "Groups the site or site-network-access belongs to.";
     }
     description
      "Grouping definition to assign
      group-ids to site or site-network-access.";
    }
    grouping site-diversity {
     container site-diversity {
       if-feature site-diversity;

       uses site-group;

       description
        "Diversity constraint type.
        All site-network-accesses will inherit the group values
        defined here.";
      }
     description
      "This grouping defines site diversity parameters.";
    }
    grouping access-diversity {
     container access-diversity {
       if-feature site-diversity;

       uses site-group;
```

```
        container constraints {
         list constraint {
          key constraint-type;

          leaf constraint-type {
           type identityref {
            base placement-diversity;
           }
           description
            "Diversity constraint type.";
          }
          container target {
           choice target-flavor {
            case id {
             list group {
              key group-id;

              leaf group-id {
               type string;
               description
                "The constraint will be applied against
                this particular group-id.";
              }
              description
               "List of groups.";
             }
            }
            case all-accesses {
             leaf all-other-accesses {
              type empty;
              description
               "The constraint will be applied against
               all other site network accesses of this site.";
             }
            }
            case all-groups {
             leaf all-other-groups {
              type empty;
              description
               "The constraint will be applied against
               all other groups managed by the customer.";
             }
            }
            description
             "Choice for the group definition.";
           }
```

```
          description
           "The constraint will be applied against
           this list of groups.";
         }
         description
          "List of constraints.";
       }
       description
        "Placement constraints for this site network access.";
     }

     description
      "Diversity parameters.";
    }
   description
    "This grouping defines access diversity parameters.";
  }

  grouping operational-requirements {
     leaf requested-site-start {
       type yang:date-and-time;
       description
        "Optional leaf indicating requested date and time when the
        service at a particular site is expected to start.";
      }

      leaf requested-site-stop {
       type yang:date-and-time;
       description
        "Optional leaf indicating requested date and time when the
        service at a particular site is expected to stop.";
      }
   description
    "This grouping defines some operational parameters.";
  }
```

```
    grouping operational-requirements-ops {
        leaf actual-site-start {
         type yang:date-and-time;
         config false;
         description
          "Optional leaf indicating actual date and time when the
          service at a particular site actually started.";
        }
        leaf actual-site-stop {
         type yang:date-and-time;
         config false;
         description
          "Optional leaf indicating actual date and time when the
          service at a particular site actually stopped.";
        }
     description
      "This grouping defines some operational parameters.";
    }

    grouping flow-definition {
     container match-flow {
      leaf dscp {
       type inet:dscp;
       description
        "DSCP value.";
      }
      leaf dot1p {
       type uint8 {
        range "0..7";
       }
       description
        "802.1p matching.";
      }
      leaf ipv4-src-prefix {
       type inet:ipv4-prefix;
       description
        "Match on IPv4 src address.";
      }
      leaf ipv6-src-prefix {
       type inet:ipv6-prefix;
       description
        "Match on IPv6 src address.";
      }
      leaf ipv4-dst-prefix {
       type inet:ipv4-prefix;
       description
        "Match on IPv4 dst address.";
      }
```

```
      leaf ipv6-dst-prefix {
       type inet:ipv6-prefix;
       description
        "Match on IPv6 dst address.";
      }
      leaf l4-src-port {
       type inet:port-number;
       description
        "Match on Layer 4 src port.";
      }
      leaf-list target-sites {
       type svc-id;
       description
        "Identify a site as traffic destination.";
      }
      container l4-src-port-range {
       leaf lower-port {
        type inet:port-number;
        description
         "Lower boundary for port.";
       }
       leaf upper-port {
        type inet:port-number;
        must ". >= ../lower-port" {
         description
          "Upper boundary must be higher than lower boundary.";
        }
        description
         "Upper boundary for port.";
       }
       description
        "Match on Layer 4 src port range.";
      }
      leaf l4-dst-port {
       type inet:port-number;
       description
        "Match on Layer 4 dst port.";
      }
      container l4-dst-port-range {
       leaf lower-port {
        type inet:port-number;
        description
         "Lower boundary for port.";
       }
```

```
       leaf upper-port {
        type inet:port-number;
        must ". >= ../lower-port" {
         description
          "Upper boundary must be higher than lower boundary.";
        }
        description
         "Upper boundary for port.";
       }
       description
        "Match on Layer 4 dst port range.";
      }
      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
      "Flow definition based on criteria.";
    }
    grouping site-service-basic {
     leaf svc-input-bandwidth {
         type uint32;
         units bps;
         description
          "From the PE's perspective, the service input
           bandwidth of the connection.";
     }
     leaf svc-output-bandwidth {
        type uint32;
        units bps;
        description
         "From the PE's perspective, the service output
          bandwidth of the connection.";
     }
```

```
     leaf svc-mtu {
      type uint16;
      units bytes;
      description
       "MTU at service level.  If the service is IP,
       it refers to the IP MTU.";
     }
     description
      "Defines basic service parameters for a site.";
    }
    grouping site-protection {
     container traffic-protection {
      if-feature fast-reroute;
      leaf enabled {
       type boolean;
       default false;
       description
        "Enables traffic protection of access link.";
      }
      description
       "Fast Reroute service parameters for the site.";
     }
     description
      "Defines protection service parameters for a site.";
    }
    grouping site-service-mpls {
     container carrierscarrier {
      if-feature carrierscarrier;
      leaf signalling-type {
       type enumeration {
        enum "ldp" {
         description
          "Use LDP as the signalling protocol
          between the PE and the CE.";
        }
        enum "bgp" {
         description
          "Use BGP (as per RFC 3107) as the signalling protocol
          between the PE and the CE.
          In this case, BGP must also be configured as
          the routing protocol.";
        }
       }
       description
        "MPLS signalling type.";
      }
```

```
      description
       "This container is used when the customer provides
       MPLS-based services.  This is used in the case of CsC.";
     }
     description
      "Defines MPLS service parameters for a site.";
    }
    grouping site-service-qos-profile {
     container qos {
      if-feature qos;
      container qos-classification-policy {
       list rule {
        key id;
        ordered-by user;

        leaf id {
         type uint16;
         description
          "ID of the rule.";
        }

        choice match-type {
         case match-flow {
          uses flow-definition;
         }
         case match-application {
          leaf match-application {
           type identityref {
            base customer-application;
           }
           description
            "Defines the application to match.";
          }
         }
         description
          "Choice for classification.";
        }

        leaf target-class-id {
         type string;
         description
          "Identification of the class of service.
          This identifier is internal to the administration.";
        }

        description
         "List of marking rules.";
       }
```

```
       description
        "Configuration of the traffic classification policy.";
      }
      container qos-profile {

       choice qos-profile {
        description
         "Choice for QoS profile.
         Can be standard profile or custom.";
        case standard {
         leaf profile {
          type string;
          description
           "QoS profile to be used.";
         }
        }
        case custom {
         container classes {
          if-feature qos-custom;
          list class {
           key class-id;

           leaf class-id {
            type string;
            description
             "Identification of the class of service.
             This identifier is internal to the administration.";
           }
           leaf rate-limit {
            type uint8;
            units percent;
            description
             "To be used if the class must be rate-limited.
             Expressed as percentage of the service bandwidth.";
           }
           container latency {
            choice flavor {
             case lowest {
              leaf use-lowest-latency {
               type empty;
               description
                "The traffic class should use the path with the
                lowest latency.";
              }
             }
```

```
            case boundary {
             leaf latency-boundary {
              type uint16;
              units msec;
              description
               "The traffic class should use a path with a
               defined maximum latency.";
             }
            }
            description
             "Latency constraint on the traffic class.";
           }
           description
            "Latency constraint on the traffic class.";
          }
          container jitter {
           choice flavor {
            case lowest {
             leaf use-lowest-jitter {
              type empty;
              description
               "The traffic class should use the path with the
               lowest jitter.";
             }
            }
            case boundary {
             leaf latency-boundary {
              type uint32;
              units usec;
              description
               "The traffic class should use a path with a
               defined maximum jitter.";
             }
            }
            description
             "Jitter constraint on the traffic class.";
           }
           description
            "Jitter constraint on the traffic class.";
          }
          container bandwidth {
           leaf guaranteed-bw-percent {
            type uint8;
            units percent;
            description
             "To be used to define the guaranteed bandwidth
             as a percentage of the available service bandwidth.";
           }
```

```
            leaf end-to-end {
             type empty;
             description
              "Used if the bandwidth reservation
               must be done on the MPLS network too.";
            }
            description
             "Bandwidth constraint on the traffic class.";
           }
           description
            "List of classes of services.";
          }
          description
           "Container for list of classes of services.";
         }

        }

       }
       description
        "QoS profile configuration.";
      }
      description
       "QoS configuration.";
     }
     description
      "This grouping defines QoS parameters for a site.";
    }

    grouping site-security-authentication {
     container authentication {
      description
       "Authentication parameters.";
     }
     description
      "This grouping defines authentication parameters for a site.";

    }
    grouping site-security-encryption {
     container encryption {
      if-feature encryption;
      leaf enabled {
       type boolean;
       default false;
       description
        "If true, access encryption is required.";
      }
```

```
      leaf layer {
       type enumeration {
        enum layer2 {
         description
          "Encryption will occur at Layer 2.";
        }
        enum layer3 {
         description
          "Encryption will occur at Layer 3.
          For example, IPsec may be used.";
        }
       }
       mandatory true;
       description
        "Layer on which encryption is applied.";
      }
      container encryption-profile {
       choice profile {
        case provider-profile {
         leaf profile-name {
          type string;
          description
           "Name of the SP profile to be applied.";
         }
        }
        case customer-profile {
         leaf algorithm {
          type string;
          description
           "Encryption algorithm to be used.";
         }
         choice key-type {
          case psk {
           leaf preshared-key {
            type string;
            description
             "Key coming from customer.";
           }
          }
          case pki {

          }
          description
           "Type of keys to be used.";
         }
        }
```

```
        description
         "Choice of profile.";
       }
       description
        "Profile of encryption to be applied.";
      }
      description
       "Encryption parameters.";
     }
     description
      "This grouping defines encryption parameters for a site.";
    }

    grouping site-attachment-bearer {
     container bearer {
      container requested-type {
       if-feature requested-type;
       leaf requested-type {
        type string;
        description
         "Type of requested bearer: Ethernet, DSL,
          Wireless, etc.  Operator specific.";
       }
       leaf strict {
        type boolean;
        default false;
        description
         "Defines whether requested-type is a preference
          or a strict requirement.";
       }
       description
        "Container for requested-type.";
      }
      leaf always-on {
       if-feature always-on;
       type boolean;
       default true;
       description
        "Request for an always-on access type.
         For example, this could mean no dial access type.";
      }
      leaf bearer-reference {
       if-feature bearer-reference;
       type string;
       description
        "This is an internal reference for the SP.";
      }
```

```
      description
       "Bearer-specific parameters.
       To be augmented.";
     }
     description
      "Defines physical properties of a site attachment.";
    }

    grouping site-routing {
     container routing-protocols {
      list routing-protocol {
       key type;

       leaf type {
        type identityref {
         base routing-protocol-type;
        }
        description
         "Type of routing protocol.";
       }

       container ospf {
        when "../type = 'ospf'" {
         description
          "Only applies when protocol is OSPF.";
        }
        if-feature rtg-ospf;
        leaf-list address-family {
         type address-family;

         description
          "Address family to be activated.";
        }
        leaf area-address {
         type yang:dotted-quad;
         description
          "Area address.";
        }
        leaf metric {
         type uint16;
         description
          "Metric of the PE-CE link.";
        }
```

```
        container sham-links {
         if-feature rtg-ospf-sham-link;
         list sham-link {
          key target-site;

          leaf target-site {
           type svc-id;
           description
            "Target site for the sham link connection.
             The site is referred to by its ID.";
          }
          leaf metric {
           type uint16;
           description
            "Metric of the sham link.";
          }
          description
           "Creates a sham link with another site.";
         }
         description
          "List of sham links.";
        }
        description
         "OSPF-specific configuration.";
       }

       container bgp {

        when "../type = 'bgp'" {
         description
          "Only applies when protocol is BGP.";
        }
        if-feature rtg-bgp;
        leaf autonomous-system {
         type uint32;
         description
          "AS number.";
        }
        leaf-list address-family {
         type address-family;

         description
          "Address family to be activated.";
        }
        description
         "BGP-specific configuration.";
       }
```

```
        container static {
         when "../type = 'static'" {
          description
           "Only applies when protocol is static.";
         }

         container cascaded-lan-prefixes {
          list ipv4-lan-prefixes {
           if-feature ipv4;
           key "lan next-hop";

           leaf lan {
            type inet:ipv4-prefix;
            description
             "LAN prefixes.";
           }
           leaf lan-tag {
            type string;
            description
             "Internal tag to be used in VPN policies.";
           }
           leaf next-hop {
            type inet:ipv4-address;
            description
             "Next-hop address to use on the customer side.";
           }
           description
            "List of LAN prefixes for the site.";
          }
          list ipv6-lan-prefixes {
           if-feature ipv6;
           key "lan next-hop";

           leaf lan {
            type inet:ipv6-prefix;
            description
             "LAN prefixes.";
           }
           leaf lan-tag {
            type string;
            description
             "Internal tag to be used in VPN policies.";
           }
           leaf next-hop {
            type inet:ipv6-address;
            description
             "Next-hop address to use on the customer side.";
           }
```

```
          description
           "List of LAN prefixes for the site.";
         }
         description
          "LAN prefixes from the customer.";
        }
        description
         "Configuration specific to static routing.";
       }
       container rip {

        when "../type = 'rip'" {
         description
          "Only applies when protocol is RIP.";
        }
        if-feature rtg-rip;
        leaf-list address-family {
         type address-family;

         description
          "Address family to be activated.";
        }

        description
         "Configuration specific to RIP routing.";
       }

       container vrrp {

        when "../type = 'vrrp'" {
         description
          "Only applies when protocol is VRRP.";
        }
        if-feature rtg-vrrp;
        leaf-list address-family {
         type address-family;

         description
          "Address family to be activated.";
        }
        description
         "Configuration specific to VRRP routing.";
       }

       description
        "List of routing protocols used on
        the site.  This list can be augmented.";
      }
```

```
      description
       "Defines routing protocols.";
     }
     description
      "Grouping for routing protocols.";
    }

    grouping site-attachment-ip-connection {
     container ip-connection {
      container ipv4 {
       if-feature ipv4;
       leaf address-allocation-type {
        type identityref {
         base address-allocation-type;
        }
        default "static-address";
        description
         "Defines how addresses are allocated.";
       }

       leaf number-of-dynamic-address {
        when "../address-allocation-type = 'provider-dhcp'" {
         description
          "Only applies when addresses are allocated by DHCP.";
        }
        type uint8;
        default 1;
        description
         "Describes the number of IP addresses the customer requires.";
       }
       container dhcp-relay {
        when "../address-allocation-type = 'provider-dhcp-relay'" {
         description
          "Only applies when provider is required to implement
          DHCP relay function.";
        }
        container customer-dhcp-servers {
         leaf-list server-ip-address {
          type inet:ipv4-address;
          description
           "IP address of customer DHCP server.";
         }
         description
          "Container for list of customer DHCP servers.";
        }
        description
         "DHCP relay provided by operator.";
       }
```

```
        container addresses {
         when "../address-allocation-type = 'static-address'" {
          description
           "Only applies when protocol allocation type is static.";
         }
         leaf provider-address {
          type inet:ipv4-address;
          description
           "Address of provider side.";
         }
         leaf customer-address {
          type inet:ipv4-address;
          description
           "Address of customer side.";
         }
         leaf mask {
          type uint8 {
           range "0..31";
          }
          description
           "Subnet mask expressed in bits.";
         }
         description
          "Describes IP addresses used.";
        }

        description
         "IPv4-specific parameters.";

       }
       container ipv6 {
        if-feature ipv6;
        leaf address-allocation-type {
         type identityref {
          base address-allocation-type;
         }
         default "static-address";
         description
          "Defines how addresses are allocated.";
        }
        leaf number-of-dynamic-address {
         when
         "../address-allocation-type = 'provider-dhcp' "+
         "or ../address-allocation-type "+
         "= 'provider-dhcp-slaac'" {
          description
           "Only applies when addresses are allocated by DHCP.";
         }
```

```
        type uint8;
        default 1;
        description
         "Describes the number of IP addresses the customer requires.";
       }
       container dhcp-relay {
        when "../address-allocation-type = 'provider-dhcp-relay'" {
         description
          "Only applies when provider is required to implement
          DHCP relay function.";
        }
        container customer-dhcp-servers {
         leaf-list server-ip-address {
          type inet:ipv6-address;
          description
           "IP address of customer DHCP server.";
         }
         description
          "Container for list of customer DHCP servers.";
        }
        description
         "DHCP relay provided by operator.";
       }
       container addresses {
        when "../address-allocation-type = 'static-address'" {
         description
          "Only applies when protocol allocation type is static.";
        }
        leaf provider-address {
         type inet:ipv6-address;
         description
          "Address of provider side.";
        }
        leaf customer-address {
         type inet:ipv6-address;
         description
          "Address of customer side.";
        }
        leaf mask {
         type uint8 {
          range "0..127";
         }
         description
          "Subnet mask expressed in bits.";
        }
        description
         "Describes IP addresses used.";
       }
```

```
       description
        "IPv6-specific parameters.";

      }
      container oam {
       container bfd {
        if-feature bfd;
        leaf enabled {
         type boolean;
         default false;
         description
          "BFD activation.";
        }

        choice holdtime {
         case profile {
          leaf profile-name {
           type string;
           description
            "Well-known SP profile.";
          }
          description
           "Well-known SP profile.";
         }
         case fixed {
          leaf fixed-value {
           type uint32;
           units msec;
           description
            "Expected holdtime expressed in msec.";
          }
         }
         description
          "Choice for holdtime flavor.";
        }
        description
         "Container for BFD.";
       }
       description
        "Defines the OAM mechanisms used on the connection.";
      }
      description
       "Defines connection parameters.";
     }
     description
      "This grouping defines IP connection parameters.";
    }
```

```
    grouping site-service-multicast {
     container multicast {
      if-feature multicast;
      leaf multicast-site-type {
       type enumeration {
        enum receiver-only {
         description
          "The site only has receivers.";
        }
        enum source-only {
         description
          "The site only has sources.";
        }
        enum source-receiver {
         description
          "The site has both sources and receivers.";
        }
       }
       default "source-receiver";
       description
        "Type of multicast site.";
      }
      container multicast-address-family {
       leaf ipv4 {
        if-feature ipv4;
        type boolean;
        default true;
        description
         "Enables IPv4 multicast.";
       }
       leaf ipv6 {
        if-feature ipv6;
        type boolean;
        default false;
        description
         "Enables IPv6 multicast.";
       }
       description
        "Defines protocol to carry multicast.";
      }
      leaf protocol-type {
       type enumeration {
        enum host {
         description
          "Hosts are directly connected to the provider network.
           Host protocols such as IGMP or MLD are required.";
        }
```

```
        enum router {
         description
          "Hosts are behind a customer router.
          PIM will be implemented.";
        }
        enum both {
         description
          "Some hosts are behind a customer router, and some others
          are directly connected to the provider network.
          Both host and routing protocols must be used.
          Typically, IGMP and PIM will be implemented.";
        }
       }
       default "both";
       description
        "Multicast protocol type to be used with the customer site.";
      }

      description
       "Multicast parameters for the site.";
     }
     description
      "Multicast parameters for the site.";
    }

    grouping site-management {
     container management {
      leaf type {
       type identityref {
        base management;
       }
       description
        "Management type of the connection.";
      }
      description
       "Management configuration.";
     }
     description
      "Management parameters for the site.";
    }
```

```
    grouping site-devices {
     container devices {
      must "/l3vpn-svc/sites/site/management/type = "+
       "'provider-managed' or "+
       "/l3vpn-svc/sites/site/management/type = "+
       "'co-managed'" {
        description
         "Applicable only for provider-managed or co-managed device.";
       }
      list device {
       key device-id;

       leaf device-id {
        type svc-id;
        description
         "Identifier for the device.";
       }
       leaf location {
        type leafref {
         path "/l3vpn-svc/sites/site/locations/"+
          "location/location-id";
        }
        description
         "Location of the device.";
       }
       container management {
        must "/l3vpn-svc/sites/site/management/type"+
         "= 'co-managed'" {
          description
           "Applicable only for co-managed device.";
         }
        leaf address-family {
         type address-family;

         description
          "Address family used for management.";
        }
        leaf address {
         type inet:ip-address;
         description
          "Management address.";
        }
        description
         "Management configuration.  Applicable only for
         co-managed device.";
       }
```

```
       description
        "Device configuration.";
      }
      description
       "List of devices requested by customer.";
     }
     description
      "Grouping for device allocation.";
    }
    grouping site-vpn-flavor {
     leaf site-vpn-flavor {
      type identityref {
       base site-vpn-flavor;
      }
      default site-vpn-flavor-single;
      description
       "Defines whether the site is, for example,
       a single VPN site or a multiVPN.";
     }
     description
      "Grouping for site VPN flavor.";
    }

    grouping site-vpn-policy {
     container vpn-policies {
      list vpn-policy {
       key vpn-policy-id;

       leaf vpn-policy-id {
        type svc-id;
        description
         "Unique identifier for the VPN policy.";
       }

       list entries {
        key id;

        leaf id {
          type svc-id;
          description
           "Unique identifier for the policy entry.";
        }
```

```
        container filter {
         choice lan {
          case prefixes {
           leaf-list ipv4-lan-prefix {
            if-feature ipv4;
            type inet:ipv4-prefix;
            description
             "List of IPv4 prefixes to be matched.";
           }
           leaf-list ipv6-lan-prefix {
            if-feature ipv6;
            type inet:ipv6-prefix;
            description
             "List of IPv6 prefixes to be matched.";
           }
          }
          case lan-tag {
           leaf-list lan-tag {
            type string;
            description
             "List of 'lan-tag' items to be matched.";
           }
          }
          description
           "Choice of ways to do LAN matching.";
         }
         description
          "If used, it permits the splitting of
          site LANs among multiple VPNs.
          If no filter is used, all the LANs will be
          part of the same VPNs with the same role.";
        }
        container vpn {
         leaf vpn-id {
          type leafref {
           path "/l3vpn-svc/vpn-services/"+
           "vpn-service/vpn-id";
          }
          mandatory true;
          description
           "Reference to an IP VPN.";
         }
```

```
        leaf site-role {
         type identityref {
          base site-role;
         }
         default any-to-any-role;
         description
          "Role of the site in the IP VPN.";
        }
        description
         "List of VPNs the LAN is associated with.";
       }
       description
        "List of entries for export policy.";
      }
      description
       "List of VPN policies.";
    }
    description
     "VPN policy.";
   }
   description
    "VPN policy parameters for the site.";
  }

  grouping site-maximum-routes {
   container maximum-routes {
    list address-family {
     key af;

     leaf af {
      type address-family;

      description
       "Address family.";
     }
     leaf maximum-routes {
      type uint32;
      description
       "Maximum prefixes the VRF can accept for this address family.";
     }
     description
      "List of address families.";
    }

    description
     "Defines 'maximum-routes' for the VRF.";
   }
```

```
     description
      "Defines 'maximum-routes' for the site.";
    }

    grouping site-security {
     container security {
      uses site-security-authentication;
      uses site-security-encryption;

      description
       "Site-specific security parameters.";
     }
     description
      "Grouping for security parameters.";
    }

    grouping site-service {
     container service {
      uses site-service-qos-profile;
      uses site-service-mpls;
      uses site-service-multicast;

      description
       "Service parameters on the attachment.";
     }
     description
      "Grouping for service parameters.";
    }

    grouping site-network-access-service {
     container service {
      uses site-service-basic;
      uses site-service-qos-profile;
      uses site-service-mpls;
      uses site-service-multicast;

      description
       "Service parameters on the attachment.";
     }
     description
      "Grouping for service parameters.";
    }
```

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

       leaf vpn-id {
        type svc-id;
        description
         "Identifies the target VPN.";
       }
       leaf local-sites-role {
        type identityref {
         base site-role;

        }
        default any-to-any-role;
        description
         "This describes the role of the
         local sites in the target VPN topology.";
       }
       description
        "List of extranet VPNs the local VPN is attached to.";
      }
      description
       "Container for extranet VPN configuration.";
     }
     description
      "Grouping for extranet VPN configuration.
      This provides an easy way to interconnect
      all sites from two VPNs.";
    }

    grouping site-attachment-availability {
     container availability {
      leaf access-priority {
       type uint32;
       default 1;
       description
        "Defines the priority for the access.
        The higher the access-priority value,
        the higher the preference of the access will be.";
      }
      description
       "Availability parameters (used for multihoming).";
     }
```

```
     description
      "Defines availability parameters for a site.";
    }

    grouping access-vpn-policy {
     container vpn-attachment {

      choice attachment-flavor {
       case vpn-policy-id {
        leaf vpn-policy-id {
         type leafref {
          path "/l3vpn-svc/sites/site/"+
          "vpn-policies/vpn-policy/"+
          "vpn-policy-id";
         }
         description
          "Reference to a VPN policy.";
        }
       }
       case vpn-id {
        leaf vpn-id {
         type leafref {
          path "/l3vpn-svc/vpn-services"+
          "/vpn-service/vpn-id";
         }
         description
          "Reference to a VPN.";
        }
        leaf site-role {
         type identityref {
           base site-role;
         }
         default any-to-any-role;
         description
          "Role of the site in the IP VPN.";
        }
       }
       mandatory true;
       description
        "Choice for VPN attachment flavor.";
      }
      description
       "Defines VPN attachment of a site.";
     }
     description
      "Defines the VPN attachment rules for a site's logical access.";
    }
```

```
    grouping vpn-svc-cfg {
     leaf vpn-id {
       type svc-id;
       description
        "VPN identifier.  Local administration meaning.";
      }
      leaf customer-name {
       type string;
       description
        "Name of the customer.";
      }
     leaf vpn-service-topology {
      type identityref {
       base vpn-topology;
      }
      default "any-to-any";
      description
       "VPN service topology.";
     }

     uses vpn-service-cloud-access;
     uses vpn-service-multicast;
     uses vpn-service-mpls;
     uses vpn-extranet;

     description
      "Grouping for VPN service configuration.";
    }

    grouping site-top-level-cfg {
     uses operational-requirements;
     uses customer-location-info;
     uses site-devices;
     uses site-diversity;
     uses site-management;
     uses site-vpn-policy;
     uses site-vpn-flavor;
     uses site-maximum-routes;
     uses site-security;
     uses site-service;
     uses site-protection;
     uses site-routing;

     description
      "Grouping for site top-level configuration.";
    }
```

```
    grouping site-network-access-top-level-cfg {
     leaf site-network-access-type {
      type identityref {
       base site-network-access-type;
      }
      default "point-to-point";
      description
       "Describes the type of connection, e.g.,
       point-to-point or multipoint.";
     }

     choice location-flavor {
      case location {
       when "/l3vpn-svc/sites/site/management/type = "+
         "'customer-managed'" {
          description
           "Applicable only for customer-managed device.";
       }
       leaf location-reference {
        type leafref {
         path "/l3vpn-svc/sites/site/locations/"+
            "location/location-id";
        }
        description
         "Location of the site-network-access.";
       }
      }
      case device {
       when "/l3vpn-svc/sites/site/management/type = "+
         "'provider-managed' or "+
         "/l3vpn-svc/sites/site/management/type = "+
         "'co-managed'" {
          description
           "Applicable only for provider-managed or co-managed device.";
       }
       leaf device-reference {
        type leafref {
         path "/l3vpn-svc/sites/site/devices/"+
            "device/device-id";
        }
        description
         "Identifier of CE to use.";
       }
      }
      mandatory true;
      description
       "Choice of how to describe the site's location.";
     }
```

```
     uses access-diversity;
     uses site-attachment-bearer;
     uses site-attachment-ip-connection;
     uses site-security;
     uses site-network-access-service;
     uses site-routing;
     uses site-attachment-availability;
     uses access-vpn-policy;

     description
      "Grouping for site network access top-level configuration.";
    }

    /* Main blocks */

    container l3vpn-svc {
     container vpn-services {
      list vpn-service {
       key vpn-id;

       uses vpn-svc-cfg;

       description
        "List of VPN services.";
      }
      description
       "Top-level container for the VPN services.";
     }

     container sites {
      list site {
       key site-id;

       leaf site-id {
        type svc-id;
        description
         "Identifier of the site.";
       }

       uses site-top-level-cfg;
       uses operational-requirements-ops;
```

```
       container site-network-accesses {
        list site-network-access {
         key site-network-access-id;

         leaf site-network-access-id {
          type svc-id;
          description
           "Identifier for the access.";
         }
         uses site-network-access-top-level-cfg;

         description
          "List of accesses for a site.";
        }
        description
         "List of accesses for a site.";
       }

       description
        "List of sites.";
      }
      description
       "Container for sites.";
     }

     description
      "Main container for L3VPN service configuration.";
    }

   }
   <CODE ENDS>
```

10. Security Considerations

The YANG module defined in this document MAY be accessed via the RESTCONF protocol [RFC8040] or the NETCONF protocol [RFC6241]. The lowest RESTCONF or NETCONF layer requires that the transport-layer protocol provide both data integrity and confidentiality; see Section 2 in [RFC8040] and Section 2 in [RFC6241]. The client MUST carefully examine the certificate presented by the server to determine if it meets the client's expectations, and the server MUST authenticate client access to any protected resource. The client identity derived from the authentication mechanism used is subject to the NETCONF Access Control Model (NACM) [RFC6536]. Other protocols that are used to access this YANG module are also required to support similar security mechanisms.

The data nodes defined in the "ietf-l3vpn-svc" YANG module MUST be carefully created, read, updated, or deleted as appropriate. The entries in the lists below include customer-proprietary or confidential information; therefore, access to confidential information MUST be limited to authorized clients, and other clients MUST NOT be permitted to access the information.

- /l3vpn-svc/vpn-services/vpn-service
- /l3vpn-svc/sites/site

The data model proposes some security parameters than can be extended via augmentation as part of the customer service request; those parameters are described in Section 6.9.

11. IANA Considerations

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

```
URI: urn:ietf:params:xml:ns:yang:ietf-l3vpn-svc
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.
```

This document adds a new YANG module name in the "YANG Module Names" registry [RFC6020]:

```
Name: ietf-l3vpn-svc
Namespace: urn:ietf:params:xml:ns:yang:ietf-l3vpn-svc
Prefix: l3vpn-svc
Reference: RFC 8049
```

12. References

12.1. Normative References

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

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

[RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, DOI 10.17487/RFC4026, March 2005, <http://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, <http://www.rfc-editor.org/info/rfc4364>.

[RFC4577] Rosen, E., Psenak, P., and P. Pillay-Esnault, "OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4577, DOI 10.17487/RFC4577, June 2006, <http://www.rfc-editor.org/info/rfc4577>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <http://www.rfc-editor.org/info/rfc4862>.

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

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

[RFC6513] Rosen, E., Ed., and R. Aggarwal, Ed., "Multicast in MPLS/BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February 2012, <http://www.rfc-editor.org/info/rfc6513>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <http://www.rfc-editor.org/info/rfc7950>.

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

12.2. Informative References

[RFC4110] Callon, R. and M. Suzuki, "A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC 4110, DOI 10.17487/RFC4110, July 2005, <http://www.rfc-editor.org/info/rfc4110>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <http://www.rfc-editor.org/info/rfc4760>.

Acknowledgements

Thanks to Qin Wu, Maxim Klyus, Luis Miguel Contreras, Gregory Mirsky, Zitao Wang, Jing Zhao, Kireeti Kompella, Eric Rosen, Aijun Wang, Michael Scharf, Xufeng Liu, David Ball, Lucy Yong, Jean-Philippe Landry, and Andrew Leu for their contributions to this document.

Contributors

The authors would like to thank Rob Shakir for his major contributions to the initial modeling and use cases.

Authors' Addresses

Stephane Litkowski
Orange Business Services

Email: stephane.litkowski@orange.com

Luis Tomotaki
Verizon

Email: luis.tomotaki@verizon.com

Kenichi Ogaki
KDDI Corporation

Email: ke-oogaki@kddi.com