             YANG Data Model for Value Added Service (VAS)
                  draft-gu-l3sm-vas-service-model-01

Abstract

   L3SM defines a YANG data model for L3VPN service model that can be
   used to configure and manage L3VPN network. This document discusses
   generic VAS model that can be applied to L3VPN network and other
   Cloud VPN networks. The YANG model provides common structure for
   various VAS service components.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

By using Network Function Virtualization (NFV), telecommunication's
networks, currently built out of customized hardware with a specific
function, can be built upon general servers with network service
functions provided by software. Services can span everything
including managed router, firewall, security (such as intrusion and
malware detection), unified communications, and application
performance management et al. Such service can be referred to as a
value added service (VAS). A customized VAS can be created and
managed by operators for customers by adding it dynamically to the
customers' data path in service chaining in conjunction.

This document defines a YANG data model for Generic Value Added
Service (VAS) configuration and operation which provides common
structure for various VAS service components, such as firewall, load
balancing et al.

This service model can be applied to L3VPN network in conjunction
with L3VPN service model defined in [draft-ietf-l3sm-l3vpn-service-
model] to configure and manage L3VPN network and other Cloud VPN
networks.

2. Conventions and Terminology

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

   The following notations are used within the data tree and carry the
   meaning as below.

  Each node is printed as:
    <status> <flags> <name> <opts> <type>

    <status> is one of:
         + for current
         x for deprecated
         o for obsolete
    <flags> is one of:
        rw for configuration data
        ro for non-configuration data
        -x for rpcs
        -n for notifications
    <name> is the name of the node

    If the node is augmented into the tree from another module, its name
    is printed as <prefix>:<name>.
    <opts> is one of:
        ? for an optional leaf or choice
        ! for a presence container
        * for a leaf-list or list
        [<keys>] for a list's keys
    <type> is the name of the type for leafs and leaf-lists

   In this document, these words will appear with that interpretation
   only when in ALL CAPS. Lower case uses of these words are not to be
   interpreted as carrying RFC-2119 significance.

2.1. Terminologies

   NFV Network Function Virtualization

   L3SM Layer 3 VPN Service Model

3. Value Added Service (VAS) service model

   A value added service is a network function provided by the service
   operator. The Value Added Service (VAS) service model is to provide
   a common understanding on what the value added network service
   includes when deployed onto network for customers.

3.1. Service data model usage

```
   +---------------------------------------------------------------+
   |                                                               |
   |                        Application                            |
   |                                                               |
   |                                                               |
   +-----------|-----------------------------------|-------------+
               |                                   |
               |                                   | VAS Service Model
               |                                   |
               |                                   |
               |                           +--|-------------------+
               |                        +-------V---------------+  |
   +-----------V---------------+        +-----------------------+  |   |
   |                          |         |                       |  |   |
   |                          |         |                       |  |   |
   |        Controller        |         |     VNF Manager       |  |----+
   |                          |         |                       | -|- +
   +-----------|--------------+         +---|-------|-----------+  |
               |                            |       |              |
               |                            |       | CLI/Netconf |.
               |                            |       |              |
               | Openflow/Ovsdb             |       |              |
               | /Netconf...                |       |              |
     +--------|-----------+                 |       |              |
     +--------V----------+|        ++++++V+++ +++V+++++++   ++++V++++++
     +------------------+||        +       + +        +   +          +
     |                 |||        +  vLB  + +  vFW   +   +  vFW    +
     |      Device     ||+        +       + +        +   +          +
     |                 |+        +++++++++++ ++++++++++   ++++++++++++
     +------------------+
```
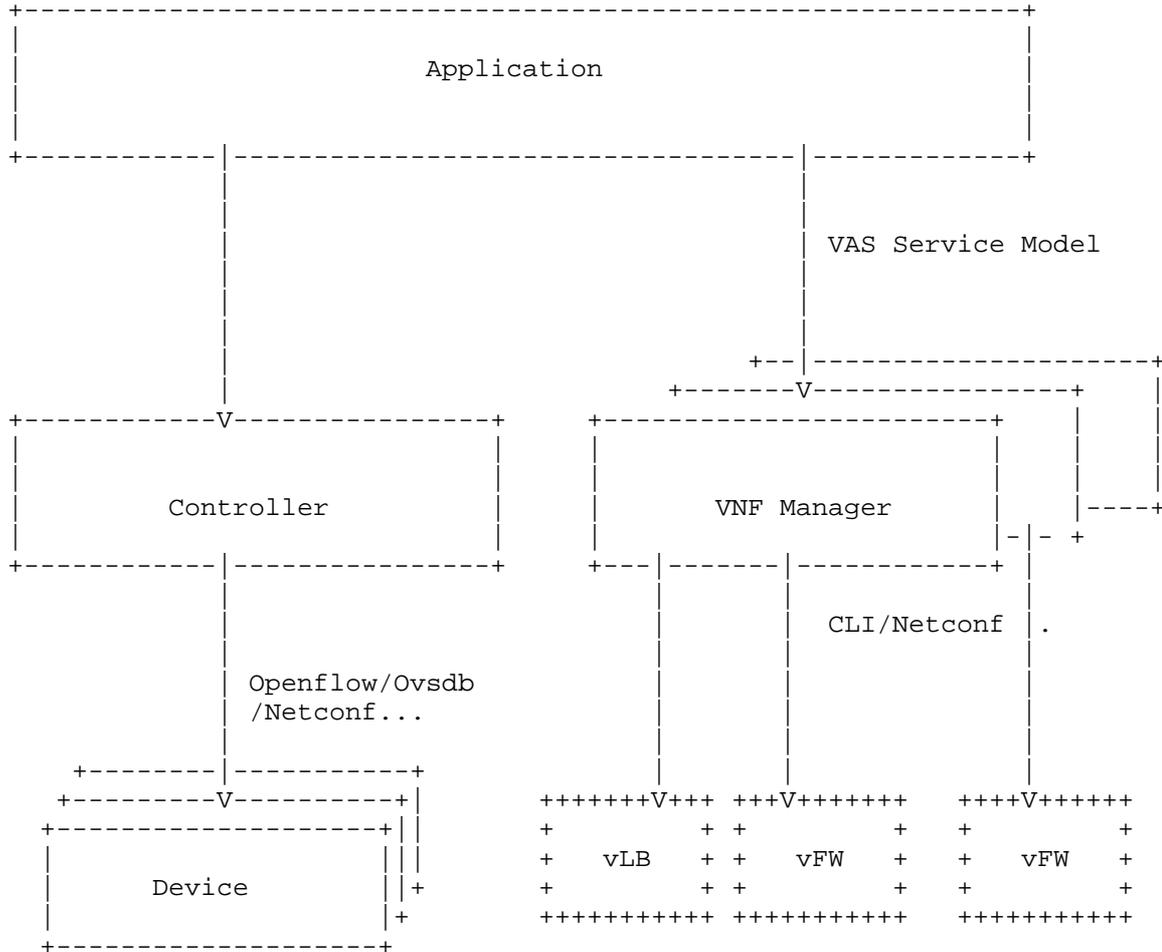
   The purpose of the VAS service model is to propose a generic
   interface to manage and deploy configuration of a VAS service
   component, for example firewall et al. A typical usage is to use
   this model as an input of Virtual Network Function Manager (VNFM)
   derived from NFV service component on application layer to configure

and manage the VNFM to create and delete virtual Load Balancing (vLB)
and virtual Firewall (vFW) instances for customers, so as to provide
value added services for tenant networks. The configuration of VNF
such as vLBs and vFWs MAY be done by CLI, or by NetConf/RestConf.

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.

4. Design of the Data Model

The YANG module includes two parts: vas-service-components and vas-
component-management. The vas-service-components defines parameters
for value added service component for a specific customer, which is
used by service providers onto VNFM to create/delete VAS service for
tenants. The vas-component-management is rps model to facilitate
issuing commands to a NETCONF server (in this case to the VNFM
component to mange a VAS) and get a response. rpc model here
abstracts specific commands for various value added services,
including load balancing and firewall services. There are two rpc
commands defined here for the management of VASs, that are component
management and policy management.

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

```
module: ietf-vas-svc
   +--rw vas-service-components
      +--rw service-component* [name]
         +--rw name                           string
         +--rw id?                            uint32
         +--rw admin-status?                  enumeration
         +--rw enable?                        boolean
         +--rw isvirtual?                     boolean
         +--rw tenant-id?                     string
         +--rw provider?                      string
         +--rw service-component-type?        identityref
         +--rw vas-policy
         |  +--rw vas-classification-policy
         |  |  +--rw rules* [id]
         |  |     +--rw id             string
         |  |     +--rw match-flow
         |  |        +--rw (type)?
         |  |           +--:(ip)
         |  |           |  +--rw (ip-version)?
         |  |           |     +--:(ipv4)
         |  |           |     |  +--rw src-ipv4-network?        inet:ipv4-prefi
x
         |  |           |     |  +--rw dst-ipv4-network?        inet:ipv4-prefi
x
         |  |           |     +--:(ipv6)
```

```
         | |              |          +--rw src-ipv6-network?        inet:ipv6-prefi
x
         | |              |          +--rw dst-ipv6-network?        inet:ipv6-prefi
x
         | |              |          +--rw flow-label?              inet:ipv6-flow-
label
         | |              |          +--rw dscp?                    inet:dscp
         | |              |          +--rw protocol?                uint8
         | |              |          +--rw source-port-range
         | |              |          |  +--rw lower-port?   inet:port-number
         | |              |          |  +--rw upper-port?   inet:port-number
         | |              |          +--rw destination-port-range
         | |              |             +--rw lower-port    inet:port-number
         | |              |             +--rw upper-port?   inet:port-number
         | |           +--:(eth)
         | |              +--rw src-mac-address?         yang:mac-address
         | |              +--rw dst-mac-address?         yang:mac-address
         | |              +--rw src-mac-address-mask?    yang:mac-address
         | |              +--rw dst-mac-address-mask?    yang:mac-address
         | +--rw std-vas-profile?         string
         +--rw availability
         |  +--rw service-type?   identityref
         +--rw management
            +--rw management-url?    string
            +--rw vas-instance-id?   uint32
            +--rw vlan-id            uint32
  rpcs:
    +---x vas-component-management
    |  +--ro input
    |  |  +--ro service-component
    |  |     +--ro name?                   string
    |  |     +--ro admin-status?           enumeration
    |  |     +--ro operation-type?         identityref
    |  |     +--ro isvirtual?              boolean
    |  |     +--ro tenant-id?              string
    |  |     +--ro provider?               string
    |  |     +--ro service-component-type?  identityref
    |  |     +--ro vas-policy
    |  |        +--ro id?    string
    |  +--ro output
    |     +--ro service-component
    |        +--ro name?                   string
    |        +--ro admin-status?           enumeration
    |        +--ro operation-type?         identityref
    |        +--ro isvirtual?              boolean
    |        +--ro tenant-id?              string
    |        +--ro provider?               string
    |        +--ro service-component-type?  identityref
    |        +--ro vas-policy
    |        |  +--ro id?    string
    |        +--ro management
```

```
      |              +--ro management-url?     string
      |              +--ro vas-instance-id?    uint32
   +---x vas-policy-management
      +--ro input
      |  +--ro service-component
      |     +--ro name?                    string
      |     +--ro isvirtual?               boolean
      |     +--ro tenant-id?               string
      |     +--ro provider?                string
      |     +--ro service-component-type?  identityref
      |     +--ro vas-policy
      |        +--ro id?    string
      +--ro output
         +--ro service-component
            +--ro name?                    string
            +--ro isvirtual?               boolean
            +--ro tenant-id?               string
            +--ro provider?                string
            +--ro service-component-type?  identityref
            +--ro vas-policy
               |  +--ro vas-classification-policy
               |  |  +--ro rules* [id]
               |  |     +--ro id               string
               |  |     +--ro match-flow
               |  |        +--ro (type)?
               |  |           +--:(ip)
               |  |           |  +--ro (ip-version)?
               |  |           |     +--:(ipv4)
               |  |           |     |  +--ro src-ipv4-network?        inet:ipv4-pr
efix
               |  |           |     |  +--ro dst-ipv4-network?        inet:ipv4-pr
efix
               |  |           |     +--:(ipv6)
               |  |           |        +--ro src-ipv6-network?        inet:ipv6-pr
efix
               |  |           |        +--ro dst-ipv6-network?        inet:ipv6-pr
efix
               |  |           |        +--ro flow-label?              inet:ipv6-fl
ow-label
               |  |           |        +--ro dscp?                    inet:dscp
               |  |           |        +--ro protocol?                uint8
               |  |           |        +--ro source-port-range
               |  |           |        |  +--ro lower-port?    inet:port-number
               |  |           |        |  +--ro upper-port?    inet:port-number
               |  |           |        +--ro destination-port-range
               |  |           |           +--ro lower-port?    inet:port-number
               |  |           |           +--ro upper-port?    inet:port-number
               |  |           +--:(eth)
               |  |              +--ro src-mac-address?          yang:mac-address
               |  |              +--ro dst-mac-address?          yang:mac-address
               |  |              +--ro src-mac-address-mask?     yang:mac-address
               |  |              +--ro dst-mac-address-mask?     yang:mac-address
               |  +--ro std-vas-profile?                string
```

```
         +--ro management
            +--ro management-url?     string
            +--ro vas-instance-id?    uint32
```

4.1. VAS service overview

   The vas-service-components top container includes generic information
   about the value added service. The name of the vas-service-
   components refers to an internal reference for the VAS, while the id
   is also the identifier of this service component used by systems.
   This identifier is purely internal to the service provider that
   offers this service. The admin-status indicates the administration
   of this value added service component. Besides, tenant-id is defined
   to indicate the customer that requires this service and presents for
   the service provider for this VAS.

4.1.1. Service component type

   The type of VAS service component is to indicate the type of service
   component, so as to indicate its virtual network function. Current
   proposal includes: firewall and loadbalancing. New VAS component
   could be added by augmentation.

4.1.2. The VAS policy

   Policies of the VAS are required for configuration, which shows the
   rules for customer traffic flows, so as to achieve their required
   value added services. The policy id refers to the set of rules
   within the configuration and management system, while the match-flow
   defines the applied traffic flows. The policies can be defined by
   service providers themselves by using policy models proposed in SUPA
   or other policy related groups.

   The std-vas-profile can also be used to show the provider standard
   vas profile to be applied. This is a reference to a well known
   profile in Service provider administration, e.g. PLATIUM for VIP.

4.1.3. Service availability

   The service availability, along with service-component-redundancy,
   shows the VAS redundancy. Within the availability container, the
   model proposes three models of redundancy: single (no redundancy
   required), primary-backup (one is primary while it goes down, the
   traffic goes to the backup component to process) and loadsharing
   (both components are used at the same time, while how to implement
   the service loadsharing is out of the scope).

Also, the availability defines four access-types to indicate the role
of the service component in the service availability system, which
includes: single-access (single component for the function), primary-
access (the primary component in the primary-backup service type),
backup-access (the backup access in the primary-backup service type)
and loadsharing-access (any access in loadsharing service type).

4.1.4. Management

The management container contains the management information of this
service component. There can use a management url to indicate where
to fetch the management script. Also a vas-instance-id is referred
to a virtual network function instance that runs.

4.1.5. rpcs

The applications can also use defined rpc commands to a NETCONF
server (in this case to the VNF manager) to configure and manage the
vas components and vas policies for customers and obtains a response.
As well, rpc here abstracts vas parameters in a technology
independent manner. The YANG module defines two rpc commands for vas
component management and vas policy management.

5. Service model usage example

As explained in section 4, this service model is intended to be
instantiated at a management layer and dispatched onto a VNF Manager
to further manage resources on network elements for value added
service components for customers. The management system serves as a
NFV orchestrator to allocate and orchestrate the required value added
services for customers.

This section provides an example on how a management system can use
this generic model to configure the required value added services for
a customer. The customer (which can also be considered as a tenant)
requires firewall service and load balancing for his private cloud
network provided by a service provider.

```
   <vas-service-components>
   <service-component>
     <name>tenant_1_fw_01</name>
     <isvirtual>true</isvirtual>
     <provider> hillstone</provider>
     <service-component-type>firewall</service-component-type>
     <vas-policy>
      <rules>
       <id> c69933c1-b472-44f9-8226-30dc4ffd454c</id>
      </rules>
     </vas-policy>
   </service-component>


   <vas-service-components>
    <service-component>
     <name>tenant_1_Pool_1</name>
     <id>8032909d-47a1-4715-90af-5153ffe39861</id>
     <isvirtual>true</isvirtual>
     <provider> hillstone</provider>
     <service-component-type>load_balance</service-component-type>
     <vas-policy>
      <rules>
       <id> c69933c1-b472-44f9-8226-30dc4ffd454c</id>
      </rules>
     </vas-policy>
   </service-component>
```

   The following XML describes the configuration of firewall service and
   load balancing service for a customer.

6. VAS YANG Module

```
<CODE BEGINS> file "ietf-vas-svc.yang"
module ietf-vas-svc {
     namespace "urn:ietf:params:xml:ns:yang:ietf-vas-svc";

     prefix vas-svc;

     import ietf-inet-types {
         prefix inet;
     }

     import ietf-yang-types {
         prefix yang;
     }

     organization
```

```
     "IETF L3SM Working Group";

   contact
       "TBD";

   description
       "The YANG module defines a generic value added service configuration
       model.";

   revision 2015-10-12 {
       description
        "TBD";
                 reference
       "draft-gu-l3sm-vas-service-model-00";}

 identity vas-service-types{
 description
 "Base identity for vas service component type.";
}
identity firewall{
 base vas-service-types;
 description
 "identity for firewall service component type.";
}
identity loadbalance{
 base vas-service-types;
 description
 "identity for firewall service component type.";
}

 identity vas-operation-type{
 description
 "Base identity for vas operation type.";
}

 identity create{
 base vas-operation-type;
 description
 "Identity for create service.";
}

 identity delete{
 base vas-operation-type;
 description
 "Identity for delete service.";
}

 identity update{
```

```
   base vas-operation-type;

  description
  "Identity for update service.";
 }

     identity service-availability {
         description
           "Base identity for site availability.";
     }

     identity loadsharing {
             base service-availability;
             description
              "Identity for loadsharing.";
     }

     identity primary  {
             base service-availability;
             description
              "Identity for primary.";
     }
     identity backup  {
             base service-availability;
             description
              "Identity for backup.";
     }

      grouping vas-availability {
         container availability {
                   leaf service-type {
                        type identityref {
                            base service-availability;
                        }
                        description
                         "Type of service availability";
                   }
                 description
                  "Service availability parameters.";
            }
         description
          "This grouping defines service availability
           parameters";
     }

 container vas-service-components{
  description
```

```
   "this container contains several value-added-service components.";

 list service-component{
  key "name";
  description
  "this container present a value-added-service component.";

  leaf name{
   type string;
   description
   "the name of value added service.";
  }

  leaf id{
   type uint32;
   description
   "the vas id";
  }

  leaf admin-status{
      type enumeration {
       enum up {
        value 1;
                 description
                 "admin status up.";
  }
        enum down {
 value 2;
                 description
                 "admin status down.";
        }
 enum testing {
        value 3;
                 description
                 "testing.";
        }
       }
       description
    "admin status";
  }

  leaf enable{
   type boolean;
   description
   "enable this vas service.";
  }

  leaf isvirtual{
```

```
     type boolean;
     description
     "if it is set to true, it indicate the vas
     is a virtual service";
    }

   leaf tenant-id{
    type string;

    description
    "tenant id";
    }

   leaf provider{
    type string;
    description
    "provider name";
    }

   leaf service-component-type{
    type identityref{
     base vas-service-types;
     }
    description
    "indicate the service component type.";
    }

   container vas-policy{
    description
        "vas policy container.";
    container vas-classification-policy{
     description
        "container of classification policy";
  list rules{
   key "id";
   description
   "list of rules";
   leaf id{
    type uint16;
    description
    "rule id";
    }

 container match-flow{
    description
    "match flow.";

    choice type{
```

```
         description
          "match flow type";

         case ip{
          choice ip-version{
        case ipv4{
         leaf src-ipv4-network{
          type inet:ipv4-prefix;
                   description
                    "source ip.";
                  }
         leaf dst-ipv4-network{

          type inet:ipv4-prefix;
          description
          "destination ip.";
         }
             description
             "ipv4";
        }
        case ipv6{
         leaf src-ipv6-network{
          type inet:ipv6-prefix;
                   description
                    "source ip.";
                  }
         leaf dst-ipv6-network{
          type inet:ipv6-prefix;
          description
          "destination ip.";
         }
         leaf flow-label{
          type inet:ipv6-flow-label;
          description
          "flow label.";
         }
         leaf dscp{
          type inet:dscp;
          description
          "Differentiated Services Code Point";
         }
         leaf protocol{
          type uint8;
          description
          "Internet Protocol number.";
         }
         container source-port-range {
                   description
```

```
              "Inclusive range representing source ports to be used.
               When only lower-port is present, it represents a single port.";
              leaf lower-port {
               type inet:port-number;
               description
               "Lower boundary for port.";
               }
        leaf upper-port {
          type inet:port-number;
          must ". >= ../lower-port" {
            error-message
            "The upper-port must be greater than or equal to lower-port";
                  description
                  "must statement.";
          }
          description
            "Upper boundary for port . If existing, the upper port
            must be greater or equal to lower-port.";
        }

              }
            container destination-port-range {
             description
             "Inclusive range representing destination ports to be used. When
              only lower-port is present, it represents a single port.";
             leaf lower-port {
              type inet:port-number;
              mandatory true;
              description
              "Lower boundary for port.";
              }
        leaf upper-port {
          type inet:port-number;
          must ". >= ../lower-port" {
            error-message
            "The upper-port must be greater than or equal to lower-port";
                  description
                  "must statement.";
          }
          description
            "Upper boundary for port . If existing, the upper port
            must be greater or equal to lower-port.";
        }
              }
      description
          "ipv6 case";
    }
        description
```

```
       "choice of ip version.";
    }
     }
    case eth{
     leaf src-mac-address{
   type yang:mac-address;
   description
   "source mac address.";
   }

   leaf dst-mac-address{
    type yang:mac-address;
    description
    "destination mac address.";
   }

   leaf src-mac-address-mask{
    type yang:mac-address;
    description
    "source mac address mask.";
   }

   leaf dst-mac-address-mask{
    type yang:mac-address;

    description
    "destination mac address mask.";
   }

     }
    }
   }
  }
    }
    leaf std-vas-profile{
  type string;
  description
  "standard vas profile.";
    }
   }
  leaf service-component-redundancy{
   type boolean;
       description
       "service component redundancy.";
  }
  uses vas-availability;
  container management{
   description
```

```
    "vas management.";
    leaf management-url{
     type string;
  description
   "management url.";
     }
    leaf vas-instance-id{
     type uint32;
  description
   "vas instance id";
    }
    leaf vlan-id{
     type uint32;
  description
   "vlan id";
    }
   }
  }
 }

   rpc vas-component-management{
    description
    "vas component management.";
    input{
     container service-component{
   description
   "service component.";

   leaf name{
    type string;
    description
    "name of service component.";
   }
    leaf admin-status{
        type enumeration {
         enum up {
          value 1;
                  description
                  "admin status up.";
    }
        enum down {
    value 2;
    description
    "admin status down.";
        }
  enum testing {
        value 3;
                  description
```

```
                  "testing";
             }
          }
                description
                "admin status.";
    }
    leaf operation-type{
     type identityref{
      base vas-operation-type;
     }
     description
     "operation type such as create, delete, update, etc.";
    }
    leaf isvirtual{
     type boolean;
     description
     "if it is set to true, it indicate the vas
     is a virtual service";
    }
    leaf tenant-id{
     type string;
     description
     "tenant identity";
    }
    leaf provider{
     type string;
     description
     "provider name";
    }
    leaf service-component-type{
     type identityref{
      base vas-service-types;
     }
     description
     "service component type.";
    }

   container vas-policy{
     description
     "value added service policy.";
     leaf id{
      type string;
  description
  "policy id.";
    }
   }
 }
   }
```

```
   output{
    container service-component{
  description
  "service component.";
  leaf name{
   type string;
   description
   "name of service component.";
  }
    leaf admin-status{
       type enumeration {
        enum up {
         value 1;
                  description
                  "admin status up.";
   }
        enum down {
   value 2;
   description
   "admin status down.";
          }
  enum testing {
        value 3;
                  description
                  "testing";
          }
         }
                 description
                 "admin status.";
   }
    leaf operation-type{
     type identityref{
      base vas-operation-type;
     }
     description
     "operation type such as create, delete, update, etc.";
    }
    leaf isvirtual{
     type boolean;
     description
     "if it is set to true, it indicate the vas
     is a virtual service";
    }
    leaf tenant-id{
     type string;

     description
     "tenant identity";
```

```
   }
   leaf provider{
    type string;
    description
    "provider name";
   }
   leaf service-component-type{
    type identityref{
     base vas-service-types;
    }
    description
    "service component type.";
   }
   container vas-policy{
    description
    "value added service policy.";
    leaf id{
     type string;
  description
  "policy id.";
    }
   }
   container management{
    description
    "vas management.";
    leaf management-url{
     type string;
  description
  "management url.";
      }
    leaf vas-instance-id{
     type uint32;
  description
  "vas instance id";
    }
   }
  }
 }
   }

   rpc vas-policy-management{
     description
 "vas policy management.";
 input{
  container service-component{
   description

   "service component.";
```

```
     leaf name{
      type string;
      description
      "name of service-component.";
     }
     leaf isvirtual{
      type boolean;
      description
      "if it is set to true, it indicate the vas
      is a virtual service";
     }

     leaf tenant-id{
      type string;
      description
      "tenant id";
     }

     leaf provider{
      type string;
      description
      "provider name";
     }

     leaf service-component-type{
      type identityref{
       base vas-service-types;
      }
      description
      "indicate the service component type.";
     }
     container vas-policy{
      description
      "value added service policy.";
      leaf id{
       type string;
    description
    "policy id.";
      }
     }
    }
   }
  }
 output{
  container service-component{
   description
   "service component.";
   leaf name{
```

```
    type string;
    description
    "name of service-component.";
   }
   leaf isvirtual{
    type boolean;
    description
    "if it is set to true, it indicate the vas
    is a virtual service";
   }

   leaf tenant-id{
    type string;
    description
    "tenant id";
   }

   leaf provider{
    type string;
    description
    "provider name";
   }

   leaf service-component-type{
    type identityref{
     base vas-service-types;
    }
    description
    "indicate the service component type.";
   }
   container vas-policy{
    description
        "vas policy.";
    container vas-classification-policy{
    description
        "vas classification policy";
  list rules{
   key "id";
   description
   "list of rules.";
   leaf id{
    type string;
    description
    "rule id";
   }

        container match-flow{
    description
```

```
     "match flow.";

     choice type{

      description
      "match flow type";

      case ip{
       choice ip-version{
     case ipv4{
      leaf src-ipv4-network{
       type inet:ipv4-prefix;
                 description
                 "source ip.";
               }
      leaf dst-ipv4-network{
       type inet:ipv4-prefix;
       description
       "destination ip.";
      }
          description
          "case of ipv4";
     }
     case ipv6{
      leaf src-ipv6-network{
       type inet:ipv6-prefix;
                 description
                 "source ip.";
               }
      leaf dst-ipv6-network{
       type inet:ipv6-prefix;
       description
       "destination ip.";
      }
      leaf flow-label{
       type inet:ipv6-flow-label;
       description
       "flow label.";
      }
      leaf dscp{
       type inet:dscp;
       description
       "Differentiated Services Code Point";
      }
      leaf protocol{
       type uint8;
       description
       "Internet Protocol number.";
```

```
      }
      container source-port-range {
              description
               "Inclusive range representing source ports to be used.
                When only lower-port is present, it represents a single port.";
              leaf lower-port {

                type inet:port-number;
                description
                "Lower boundary for port.";
                }
                  leaf upper-port {
        type inet:port-number;
        must ". >= ../lower-port" {
          error-message
          "The upper-port must be greater than or equal to lower-port";
                  description
                  "must statement.";
        }
        description
          "Upper boundary for port . If existing, the upper port
          must be greater or equal to lower-port.";
      }
                }
            container destination-port-range {
             description
              "Inclusive range representing destination ports to be used. When
               only lower-port is present, it represents a single port.";
              leaf lower-port {
               type inet:port-number;
               mandatory true;
               description
               "Lower boundary for port.";
               }
      leaf upper-port {
         type inet:port-number;
         must ". >= ../lower-port" {
           error-message
           "The upper-port must be greater than or equal to lower-port";
                   description
                   "must statement.";
         }
         description
           "Upper boundary for port . If existing, the upper port
           must be greater or equal to lower-port.";
      }
               }
         description
```

```
        "case of ipv6";
   }
       description
       "choice of ip version.";
  }
 description
 "case of ip.";
   }
   case eth{
    leaf src-mac-address{
 type yang:mac-address;
 description
 "source mac address.";
 }

 leaf dst-mac-address{
  type yang:mac-address;

  description
  "destination mac address.";
 }

 leaf src-mac-address-mask{
  type yang:mac-address;
  description
  "source mac address mask.";
 }

 leaf dst-mac-address-mask{
  type yang:mac-address;
  description
  "destination mac address mask.";
 }
  description
      "case of ethernet";
  }
 }
 }
}
   }
   leaf std-vas-profile{
 type string;
 description
 "standard vas profile.";
   }
  }
 container management{
  description
```

```
     "vas management.";
     leaf management-url{
      type string;
   description
   "management url.";
       }
     leaf vas-instance-id{
      type uint32;
   description
   "vas instance id";
      }
     }
    }
   }
  }
 }
}
<CODE ENDS>
```

7. Security Considerations

   TBC.

8. IANA Considerations

   TBC.

9. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", March 1997.

Authors' Addresses

   Rong Gu
   China Mobile
   32 Xuanwumen West Ave, Xicheng District
   Beijing 100053
   China

   Email: gurong@chinamobile.com

Chen Li
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing 100053
China

Email: lichenyj@chinamobile.com

Yan Zhuang
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: zhuangyan.zhuang@huawei.com

Zitao Wang
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: wangzitao@huawei.com

                 YANG Data Model for L3VPN service delivery
                    draft-ietf-l3sm-l3vpn-service-model-19

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. The document is limited
   to the BGP PE-based VPNs as described in [RFC4026], [RFC4110] and
   [RFC4364]. This model is intended to be instantiated at management
   system to deliver the overall service. This model is not a
   configuration model to be used directly on network elements. This
   model provides an abstracted view of the Layer 3 IPVPN service
   configuration components. It will be up to a management system to
   take this as an input and use specific configurations models to
   configure the different network elements to deliver the service. How
   configuration of network elements is done is out of scope of the
   document.

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].

Status of This Memo

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

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

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

   This Internet-Draft will expire on May 8, 2017.

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 be used also as input to automated
   control and configuration applications.

1.1. Terminology

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

   o client

   o configuration data

   o server

   o state data

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

o augment

o data model

o data node

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

This document presents some configuration examples using XML representation.

1.2. Tree diagram

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

The meaning of the symbols in these diagrams is as follows:

o Brackets "[" and "]" enclose list keys.

o Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.

o Abbreviations before data node names: "rw" means configuration (read-write), and "ro" state data (read-only).

o Symbols after data node names: "?" means an optional node and "*" denotes a "list" or "leaf-list".

o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

o Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Acronyms

AAA: Authentication, Authorization, Accounting.

ACL: Access Control List.

ASM: Any-Source Multicast.

BFD: Bidirectional Forwarding Detection.

BGP: Border Gateway Protocol.

CE: Customer Edge.

CLI: Command Line Interface.

CsC: Carrier's Carrier.

CSP: Cloud Service Provider.

DHCP: Dynamic Host Configuration Protocol.

IGMP: Internet Group Management Protocol.

LAN: Local Area Network.

MLD: Multicast Listener Discovery.

MTU: Maximum Transmission Unit.

NAT: Network Address Translation.

NNI: Network to Network Interface.

OAM: Operation Administration and Management.

OSPF: Open Shortest Path First.

OSS: Operations Support System.

PE: Provider Edge.

POP: Point Of Presence.

PIM: Protocol Independent Multicast.

QoS: Quality Of Service.

RIP: Routing Information Protocol.

RD: Route Distinguisher.

RP: Rendez-vous Point.

RT: Route Target.

SLA: Service Level Agreement.

SLAAC: Stateless Address AutoConfiguration.

SP: Service Provider.

SSM: Source-Specific Multicast.

VPN: Virtual Private Network.

VRF: VPN Routing and Forwarding.

VRRP: Virtual Router Redundancy Protocol.

3. Definitions

Customer Edge (CE) Device: Equipment that is dedicated to a
particular customer and 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: Equipment managed by the Service Provider
(SP) that 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 on 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 IPVPN 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 on 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 [RFC4110] and [RFC4364].

5. Service data model usage

```
          L3VPN-SVC |
             MODEL  |
                    |
            +------------------+          +-----+
            |   Orchestration  | < --- > | OSS |
            +------------------+          +-----+
                 |          |
            +----------------+    |
            | Config manager |    |
            +----------------+    |
                 |          |
                 | Netconf/CLI ...
                 |          |
          +-----------------------------------------------+
                         Network

                 +++++++
                 + AAA +
                 +++++++

   ++++++++  Bearer ++++++++          ++++++++       ++++++++
   + CE A + ------- + PE A +          + PE B + ---- + CE B +
   ++++++++  Cnct   ++++++++          ++++++++       ++++++++

   Site A                                      Site B
```
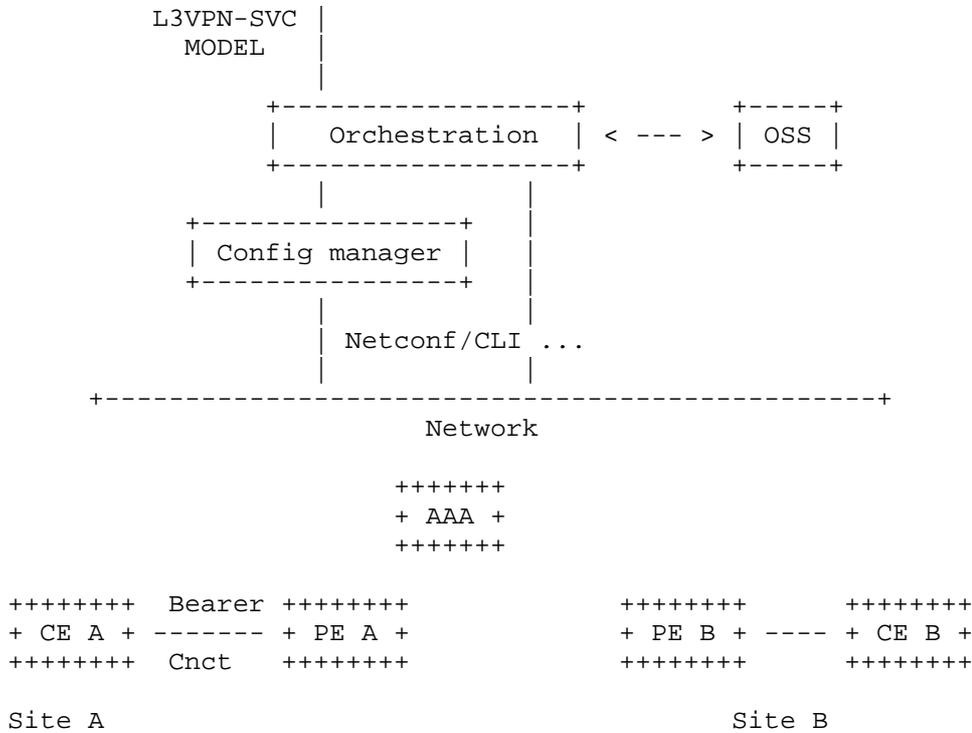
   The idea of the L3 IPVPN service model is to propose an abstracted
   interface between customers and network operators to manage
   configuration of components of a L3VPN service. A typical usage is
   to use this model as an input for an orchestration layer who will be
   responsible to translate it to orchestrated configuration of network
   elements who will be part of the service. The network elements can
   be routers, but also servers (like AAA), and not limited to these
   examples. The configuration of network elements can be done by the
   CLI, or by NETCONF ([RFC6241])/RESTCONF ([I-D.ietf-netconf-restconf])
   coupled with specific configuration YANG data models (BGP, VRF, BFD
   ...) or any other way.

   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 in two main containers : vpn-services,
   sites.

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

   A site is composed of at least one site-network-access and may have
   multiple site-network-access in case of multihoming. 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 layer 3
   protocol oriented properties. The bearer may be allocated
   dynamically by the service provider and the customer may provide some
   constraints or parameters to drive the placement.

   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 describe 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 implements a lot of features allowing implementations to be
   modular. As example, an implementation may support only IPv4 VPNs
   (ipv4 feature), IPv6 (ipv6 feature), or both (by advertising both
   features). The routing protocols proposed to the customer may also
   be enabled through features. This model proposes also some features
   for more advanced options like : extranet-vpn support
   (Section 6.2.4), site diversity (Section 6.6), 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 the IP address
assignment, if those options are not filling all requirements, new
options can be added through augmentation.

6.2. VPN service overview

A vpn-service list item contains generic informations about the VPN
service. The vpn-id of the vpn-service refers to an internal
reference for this VPN service, while 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.
Current proposal 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 by augmentation.
By default, any-to-any VPN service topology is used.

6.2.1.1. Route Target allocation

Layer 3 PE-based VPN is built using route-targets as described in
[RFC4364]. It is expected the management system to allocate
automatically a set of route-targets upon a VPN service creation
request. How the management system allocates route-targets is out of
scope of the document but multiple ways could be envisaged as
described below.
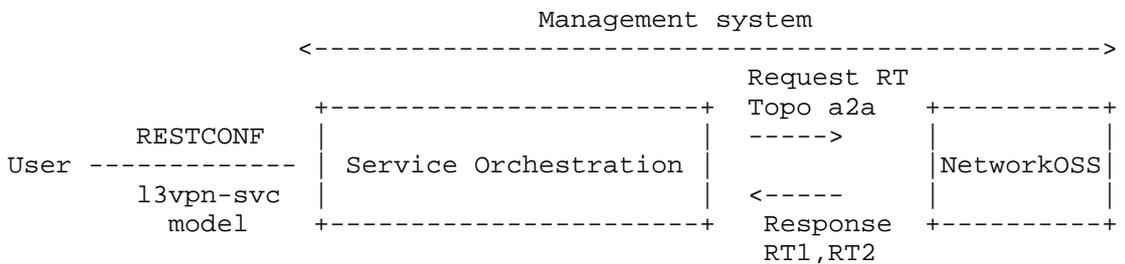
```
                         Management system
              <--------------------------------------------------->
                                       Request RT
              +----------------------+ Topo a2a   +----------+
      RESTCONF |                      |  ----->    |          |
User ----------| Service Orchestration |            |NetworkOSS|
      l3vpn-svc|                      |  <-----     |          |
        model  +----------------------+  Response  +----------+
                                         RT1,RT2
```

In the example above, a service orchestration, owning the
instantiation of this service model, request route-targets to the
network OSS. Based on the requested VPN service topology, the
network OSS replies with one or multiple route-targets. The
interface between this service orchestration and network OSS is out
of scope of this document.
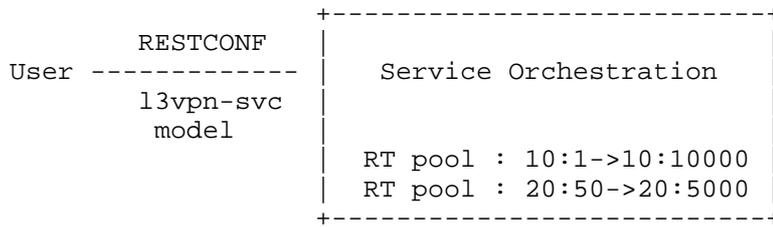
```
                    +---------------------------+
         RESTCONF   |                           |
User ------------   |    Service Orchestration  |
       l3vpn-svc    |                           |
         model      |                           |
                    |  RT pool : 10:1->10:10000 |
                    |  RT pool : 20:50->20:5000 |
                    +---------------------------+
```

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

The mechanism displayed above are just examples and should not be
considered as an exhaustive list of solutions.

6.2.1.2. Any to any

```
+------------------------------------------------------------+
|   VPN1_Site1 ------ PE1                PE2 ------ VPN1_Site2 |
|                                                            |
|   VPN1_Site3 ------ PE3                PE4 ------ VPN1_Site4 |
+------------------------------------------------------------+
```

            Figure - Any-to-any VPN service topology

In the any-to-any VPN service topology, all VPN sites can communicate
between each other without any restriction. It is expected that the
management system that receives an any-to-any IPVPN service request
through this model needs to assign and then configure the VRF and
route-targets on the appropriate PEs. In the any-to-any case, in
general a single route-target is required and every VRF imports and
exports this route-target.

6.2.1.3. Hub and Spoke

```
+--------------------------------------------------------------+
|   Hub_Site1 ------ PE1                PE2 ------ Spoke_Site1 |
|                         +-------------------------------+    |
|                         |                               |    |
|                         +-------------------------------+    |
|   Hub_Site2 ------ PE3                PE4 ------ Spoke_Site2 |
+--------------------------------------------------------------+
```
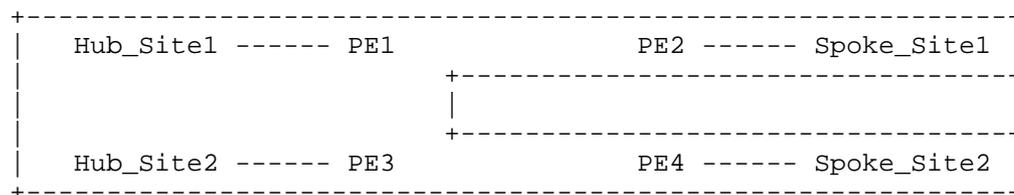
            Figure - Hub and Spoke VPN service topology

In the hub and spoke VPN service topology, all spoke sites can
communicate only with Hub sites but not between each other, and hubs
can also communicate between each other. It is expected that the
management system that owns an any to any IPVPN service request
through this model, needs to assign and then configure the VRF and
route-targets on the appropriate PEs. In the hub and spoke case, in
general a two route-targets are required (one route-target for Hub
routes, one route-target for spoke routes). A Hub VRF, connecting
Hub sites, will export Hub routes with Hub route-target, and will
import Spoke routes through Spoke route-target. It will also import
the Hub route-target to allow Hub to Hub communication. A Spoke VRF,
connecting Spoke sites, will export Spoke routes with Spoke route-
target, and will import Hub routes through Hub route-target.

The management system MUST take into account Hub and Spoke
connections constraints. 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 displayed in the figure below.

```
   Hub_Site ------- (VRF_Hub)  PE1
                               (VRF_Spoke)
                                 /  |
Spoke_Site1 ------------------+    |
                                   |
Spoke_Site2 ----------------------+
```

6.2.1.4. Hub and Spoke disjoint

```
   +--------------------------------------------------------------+
   |   Hub_Site1 ------ PE1              PE2 ------ Spoke_Site1 |
   +------------------------+  +----------------------------+
                            |  |
   +------------------------+  +----------------------------+
   |   Hub_Site2 ------ PE3              PE4 ------ Spoke_Site2 |
   +--------------------------------------------------------------+
```
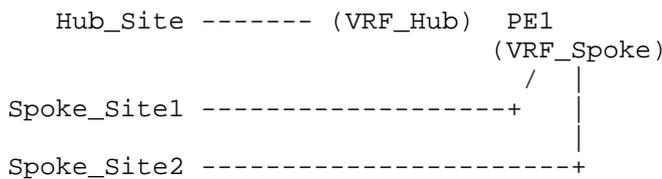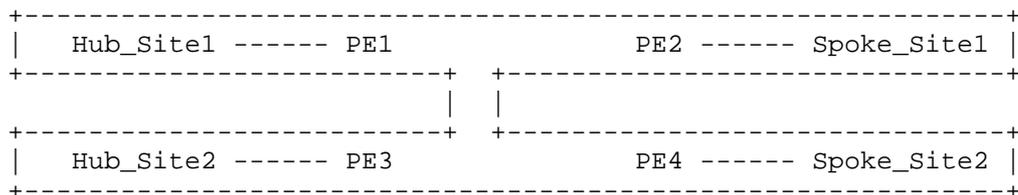
        Figure - Hub and Spoke disjoint VPN service topology

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

   routes through Spoke route-target. A Spoke VRF, connecting Spoke
   sites, will export Spoke routes with Spoke route-target, and will
   import Hub routes through Hub route-target.

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

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

6.2.2. Cloud access

   The proposed model provides a cloud access configuration through the
   cloud-access container. The usage of cloud-access is targeted for
   public cloud. An Internet access can also be considered as a public
   cloud access service. The cloud-access container provides parameters
   for network address translations 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, in case the
   customer is providing the IP address to be used for the cloud access.
   If the service provider is providing this address, the customer-
   address is not necessary as it can be picked from a service provider
   pool.

   By default, all sites in the IPVPN MUST be authorized to access to
   the cloud. In case restrictions are required, a user MAY configure
   the permit-site or deny-site leaf-list. The "permit-site" defines
   the list of sites authorized for cloud access. The "deny-site"
   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 of this document.

```
                    IPVPN
       ++++++++++++++++++++++++++++++++      ++++++++++++
       +              Site 3         + --- +  Cloud1 +
       + Site 1                      +      ++++++++++++
       +                             +
       + Site 2                      + --- +++++++++++++
       +                             +      + Internet +
       +              Site 4         +      +++++++++++++
       ++++++++++++++++++++++++++++++++
                      |
                ++++++++++
                + Cloud2 +
                ++++++++++
```
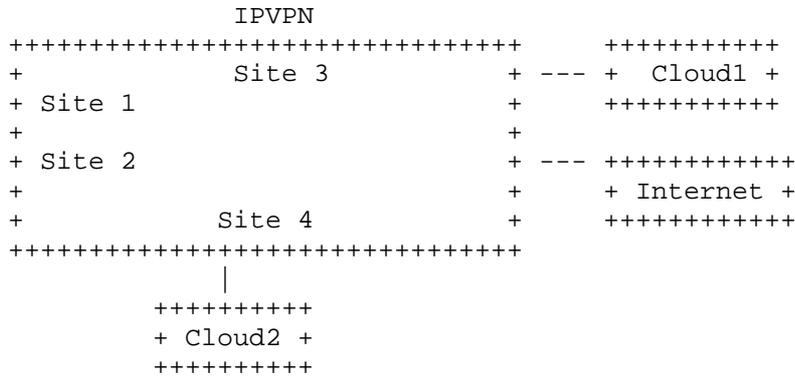
   In the example above, we may configure the global VPN to access
   Internet by creating a cloud-access pointing to the cloud identifier
   for the Internet service. No authorized-sites will be configured as
   all sites are required to access the Internet. The "address-
   translation/nat44/enabled" 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 Site1 and Site2 requires access to Cloud1, a new cloud-access will
   be created pointing to the cloud identifier of Cloud1. The "permit-
   site" leaf-list will be filled with a reference to Site1 and Site2.

```
    <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 Site1 requires access to Cloud2, a new cloud-
access will be created pointing to the cloud identifier of Cloud2.
The "deny-site" leaf-list will be filled with a reference to Site1.

```
    <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 VPN is described in [RFC6513].

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

   The user of this model will need to fill the flavor of trees that
   will be used by customer within the IPVPN (Customer tree). The
   proposed model supports bidirectional, shared and source-based trees
   (and can be augmented). Multiple flavors of tree can be supported
   simultaneously.

```
                            Operator network
                         _____
                       /                   \
                      |                     |
                      |                     |
                      (SSM tree)            |
      Recv (IGMPv3) -- Site2 ------- PE2    |
                      |                      PE1 --- Site1 --- Source1
                      |                     |    |          \
                      |                     |    |           -- Source2
                      |                     |    |
                      (ASM tree)            |
      Recv (IGMPv2) -- Site3 ------- PE3    |
                      |                     |
                      (SSM tree)            |
      Recv (IGMPv3) -- Site4 ------- PE4    |
                      | /                   |
      Recv (IGMPv2) -- Site5 --------       |
                      (ASM tree)            |
                      |                     |
                       _____/
```
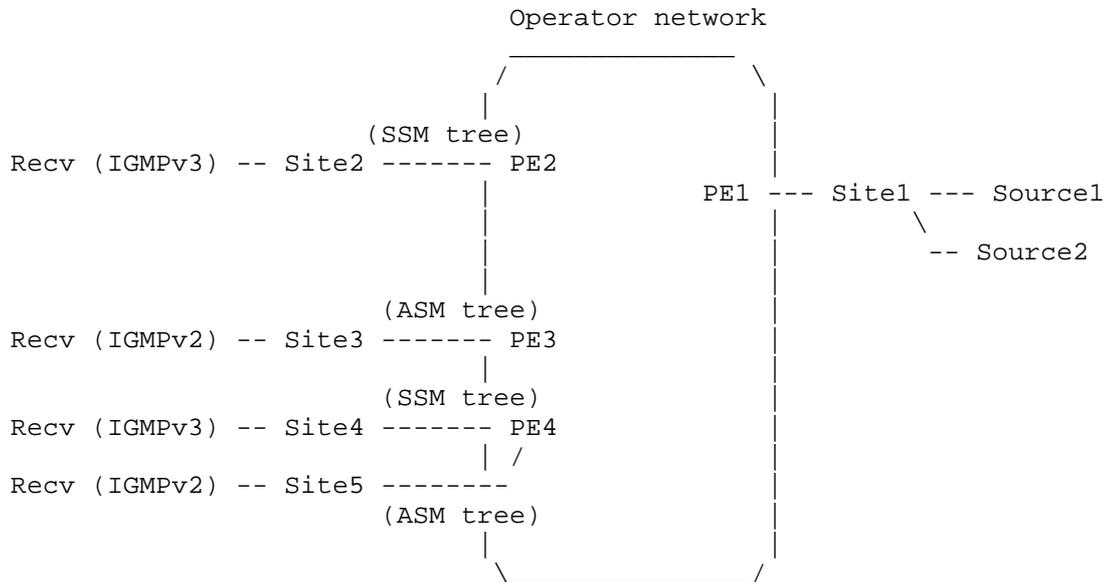
   In case of an ASM flavor requested, this model requires to fill the
   rp and rp-discovery parameters. Multiple RP to group mappings can be
   created using the rp-group-mappings container. For each mapping, the
   RP service can be managed by the service provider using the leaf
   "provider-managed/enabled" set to true. In case of provider managed
   RP, the user can request a rendez-vous point redundancy and/or an
   optimal traffic delivery. Those parameters will help the service
   provider to select the appropriate technology or architecture to
   fulfill the customer service requirement: for instance, in case of a
   request for an optimal traffic delivery, a service provider may use
   Anycast-RP or RP-tree to SPT switchover architectures.

   In 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.

   User can define a specific rp-discovery mechanism like: auto-rp,
   static-rp, bsr-rp modes. By default, the model considers 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
   and customer managed RPs. In case of a provider managed RP, if the
   user wants to use bsr-rp as a discovery protocol, a service provider
   should consider the provider managed rp-group-mappings for the bsr-rp
   configuration. The service provider will then configure its selected
   RPs to be bsr-rp-candidates. In case of a customer managed RP and a
   bsr-rp discovery mechanism, the rp-address provided will be
   considered as bsr-rp candidate.

6.2.4. Extranet VPNs

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

                +----------+   +----------+
               /   \   /   \
      SiteA -- | VPN A | --- | VPN B | --- SiteB
               \   /   \   / (Shared
                +----------+   +----------+   resources)

   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 VPN connection scenario can be achieved by the VPN policy
   defined in Section 6.5.2.2. But there are some simple cases where a
   particular VPN (VPN A) needs to access to all resources in a VPN B.
   The model provides an easy way to setup 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" must be used on customer VPNs
   accessing extranet resources in another VPN. In the figure above, in
   order to give access for VPN A to VPN B, extranet-vpns container
   needs to be configured under VPN A with an entry corresponding to VPN
   B and 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 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 to 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 between 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 more complex VPN interconnection scenario (e.g. only part of sites of VPN A accessing only part of sites of VPN B) needs to be achieved using the vpn attachment defined in Section 6.5.2 and especially the VPN policy 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.

```
                                              +------------+
                                             /              \
  +------------------+              +-----|      VPN1       |
  |                  |              |      \              /
  |  New York Office | ----- (site) -----+       +------------+
  |                  |              |       +------------+
  +------------------+              |      /              \
                                    +-----|      VPN2       |
                                           \              /
                                            +------------+
```
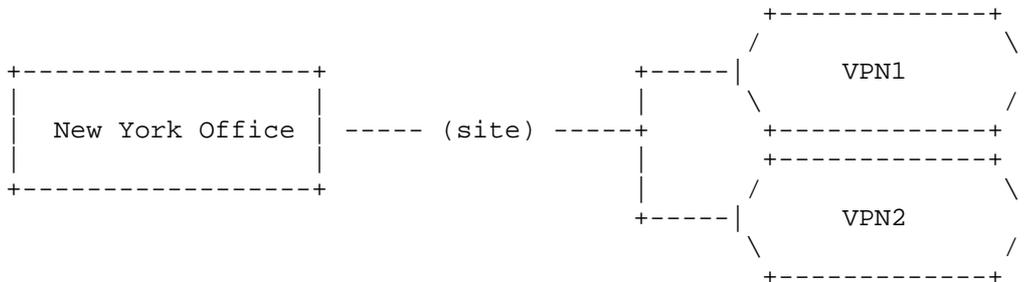
A site is composed of some characteristics :

o Unique identifier (site-id): to uniquely identify the site within the overall network infrastructure. The identifier is a string allowing to any encoding for the local administration of the VPN service.

   o Locations (locations): site location information to allow easy
      retrieval on nearest available resources. A site may be composed
      of multiple locations.

   o Devices: the customer can request one or more customer premise
      equipments from the service provider for a particular site.

   o Management (management): defines the model of management of the
      site, for example : co-managed, customer managed or provider
      managed.

   o Site network accesses (site-network-accesses): defines the list of
      network accesses associated to 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.

      +------------------+   Site
      | |--------------------------------
      | |****** (site-network-access#1) ******
      | New York Office |
      | |****** (site-network-access#2) ******
      | |--------------------------------
      +------------------+
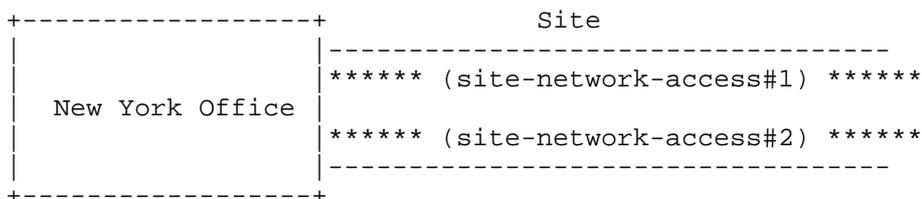
   Multiple site-network-accesses are used for instance in case of
   multihoming. Some other meshing cases may also involve multiple
   site-network-accesses.

   The site configuration is viewed as a global entity, we assume that
   it is mostly the role of the management 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 multilocation site is an headquarter 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.


```
     New York Site


   +-----------------+              Site
   | +-------------+ |---------------------------------
   | | Manhattan   | |****** (site-network-access#1) ******
   | +-------------+ |
   | +-------------+ |
   | | Brooklyn    | |****** (site-network-access#2) ******
   | +-------------+ |
   |                 |---------------------------------
   +-----------------+
```
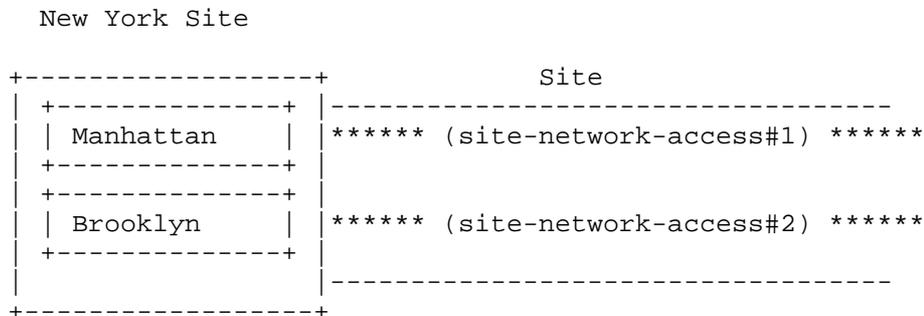

A customer may also request some premise equipments (CEs) to the
service provider through the "devices" container. Requesting a CE
implies a provider-managed or co-managed model. A particular device
must be ordered to a particular already configured location. This
would help the service provider to send the device to the appropriate
postal address. In a multilocation 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...

6.3.2. Site network accesses

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

o bearer: defines requirements of the attachment (below Layer 3).

o connection: defines Layer 3 protocol parameters of the attachment.

o availability: defines the site availability policy. The
   availability parameters are defined in Section 6.7

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

   o point-to-point: describes a point to point connection between the
      service provider and the customer.

   o multipoint: describes a multipoint connection between the service
      provider and the customer.

   The type of site-network-access may have an impact on the parameters
   offered to the customer, e.g., a service provider may not offer
   encryption for multipoint accesses. Deciding what parameter is
   supported for point-to-point and/or multipoint accesses is up to the
   provider and is out of scope of this document. Some containers
   proposed in the model may require extension 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 to 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 the PE-CE addressing or CE to customer LAN addressing. In
   any case, it describes the provider to customer responsibility
   boundary. 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 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 of IP address
   allocation:

   o provider-dhcp: the provider will provide DHCP service for customer
      equipments, this is can be applied to either IPv4 and IPv6
      containers.

   o provider-dhcp-relay: the provider will provide DHCP relay service
      for customer equipments, this is applicable to both IPv4 and IPv6

addressing. The customer needs to fill DHCP server list to be
used.

o static-address: Addresses will be assigned manually, this is
   applicable to both IPv4 and IPv6 addressing.

o slaac: enables stateless address autoconfiguration ([RFC4862]).
   This is applicable only for IPv6.

o provider-dhcp-slaac: the provider will provide DHCP service for
   customer equipments as well as stateless address
   autoconfiguration. This is applicable only for IPv6.

In the dynamic addressing mechanism, it is expected from the service
provider 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 by
augmentation. The provider can propose some profiles to the customer
depending of the service level the customer wants to achieve.
Profile names must be communicated to the customer. This
communication is out of scope of this document. Some fixed values
for the holdtime period may also be imposed by the customer if the
provider enables it.

The OAM container can easily be augmented by other mechanisms,
especially work from LIME Working Group may be reused.

6.3.2.3. Inheritance of parameters between site and site-network-access

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

In terms of provisionning impact, it will be up to the implementation
to decide of the appropriate behavior when modifying existing
configurations. But the service provider will need to communicate to
the user about the impact of using inheritance. For example, if we
consider that a site has already provisionned three site-network-
accesses, what will happen if customer is changing a service
parameter at site level ? An implementation of this model may update

the service parameters of all already provisionned 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
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 is any-to-any-role.

In the hub-spoke or hub-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 flavor defines
the way the VPN multiplexing is done. The current version of the
model supports four flavors:

o site-vpn-flavor-single: the site belongs to only one VPN.

o site-vpn-flavor-multi: the site belongs to multiple VPNs and all
   the logical accesses of the sites belongs to the same set of VPNs.

o 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).

o 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 the single VPN attachment. The site
connects to only one VPN.

```
                                           +--------+
+-----------------+              Site      /          \
|                 |------------------------|          |
|                 |***(site-network-access#1)***|  VPN1    |
|  New York Office |                        |          |
|                 |***(site-network-access#2)***|      |
|                 |------------------------|          |
+-----------------+                        \          /
                                           +--------+
```
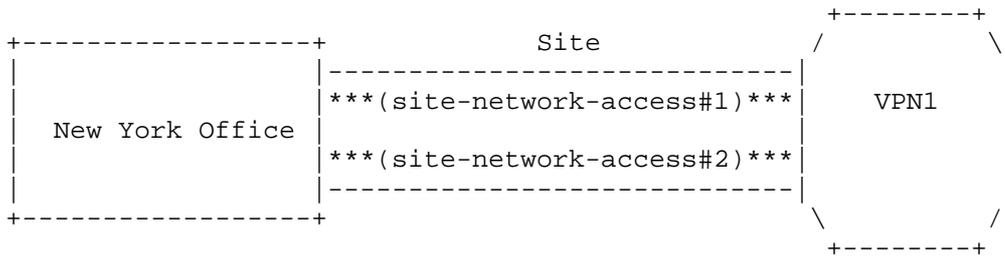
6.5.1.2. Multi VPN attachment : site-vpn-flavor-multi

   The figure below describes a site connected to multiple VPNs.

```
                                              +---------+
                                   +---/----+      \
+-----------------+              Site      /   |       \ |VPN B|
|                 |-------------------------- |        |      |
|                 |***(site-network-access#1)******* |      |      |
|  New York Office |                      |   |      |      |
|                 |***(site-network-access#2)******* \      |   /
|                 |------------------------| VPN A+-----|---+
+-----------------+                        \          /
                                           +--------+
```
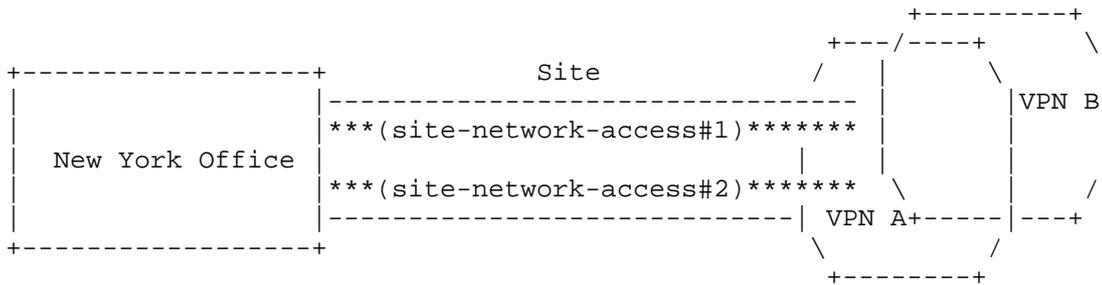
   In the example above, the New York office is multihomed, both logical
   accesses are using the same VPN attachment rules. Both logical
   accesses are connected to VPN A and VPN B.

   Reaching VPN A or VPN B from New York office will be based on
   destination based routing. Having the same destination reachable
   from the two VPNs may cause routing troubles. This would be the role
   of the customer administration to ensure the appropriate mapping of
   its prefixes in each VPN.

6.5.1.3. Sub VPN 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 VPN. A typical use case of subVPN is a customer site used by
   multiple affiliates with private resources for each affiliates that
   cannot be shared (communication is prevented between the affiliates).
   It is similar than having separate sites instead that the customer
   wants to share some physical components while keeping a strong
   communication isolation between affiliates. In the example, the
   access#1 is attached to VPN B while the access#2 is attached to VPNA.

```
+-----------------+        Site          +--------+
|                 |------------------------------/      \
|                 |****(site-network-access#1)******|   VPN B  |
|  New York Office|                         \      /
|                 |                          +--------+
|                 |                          +--------+
|                 |                          /      \
|                 |****(site-network-access#2)******|   VPN A  |
|                 |                          \      /
|                 |                          +--------+
|                 |-------------------------------
+-----------------+
```

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

```
+-----------------+        Site          +-----+
|                 |------------------------------/     +----+
|                 |****(site-network-access#1)******| VPN B/       \
|  New York Office|                         \     | VPN C  |
|                 |                          +----\       /
|                 |                                +-----+
|                 |
|                 |                          +------+
|                 |                          /     +-----+
|                 |****(site-network-access#2)******|  VPN A/       \
|                 |                          \     | VPN D  |
|                 |                          +------\       /
|                 |-------------------------------      +---+
+-----------------+
```

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

```
  +-----------------+            Site                 +-----+
  |                 |--------------------------------/    +----+
  |                 |****(site-network-access#1)******| VPNB /       \
  |  New York Office|****(site-network-access#2)******\     | VPN C  |
  |                 |                                 +----\       /
  |                 |                                      +-----+
  |                 |
  |                 |
  |                 |                                +------+
  |                 |                                /    +-----+
  |                 |****(site-network-access#3)******|  VPNA /      \
  |                 |****(site-network-access#4)****** \     | VPN D |
  |                 |                                +------\      /
  |                 |--------------------------------        +---+
  +-----------------+
```

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

6.5.1.4. NNI : site-vpn-flavor-nni

   Some Network to Network Interface (NNI) scenario may be modeled using
   the site container (see Section 6.15.1). Using the site container to
   model an NNI is only one possible option for NNI (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 service provider to
   identify that the requested VPN connection is not a regular site but
   a NNI as specific default device configuration parameters may be
   applied in case of NNI (e.g. ACLs, routing policies...).

```
        SP A                                      SP B
  --------------------                    --------------------
 /                    \                  /                    \
|                    |  |              |  |                    |
|         ++++++++ InterAS link ++++++++                      |
|         +       +_____ +       +                     |
|         + (VRF1)--(VPN1)----(VRF1)    +                     |
|         + ASBR +              + ASBR  +                     |
|         + (VRF2)--(VPN2)----(VRF2)    +                     |
|         +       +_____+        +                     |
|         ++++++++              ++++++++                      |
|            |                     |                          |
|            |                     |                          |
|            |                     |                          |
|         ++++++++ InterAS link ++++++++                      |
|         +       +_____ +       +                     |
|         + (VRF1)--(VPN1)----(VRF1)    +                     |
|         + ASBR +              + ASBR  +                     |
|         + (VRF2)--(VPN2)----(VRF2)    +                     |
|         +       +_____+       +                     |
|         ++++++++              ++++++++                      |
|            |                     |                          |
|            |                     |                          |
 \          /                      \                         /
  --------------------              --------------------
```

   The figure above describes an option A NNI scenario that can be
   modeled using the site container. In order to connect its customer
   VPN (VPN1 and VPN2) on the SP B network, 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
   IPVPN 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 of attachment:

   o By referencing directly the target VPN.

   o By referencing a VPN policy for more complex attachments.

   A choice is implemented to allow user to choose the best fitting
   flavor.

6.5.2.1. Reference 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 case of single VPN
   attachment or subVPN with single VPN attachment per logical access.
   When referencing a vpn-id, the site-role 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-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 helps to 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 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.

```
+---------------------------------------------------------------+
|        Site1 ------ PE7                                        |
+-----------------------+              [VPN2]                    |
                        |                                        |
+-----------------------+                                        |
|        Site2 ------ PE3            PE4 ------ Site3             |
+-------------------------------+                                |
                                |                                |
+-----------------------------------------------------------+   |
|        Site4 ------ PE5        |  PE6 ------ Site5          |   |
|                                |                           |   |
|                      [VPN3]                                |   |
+-----------------------------------------------------------+   |
                                |                                |
                        +-------------------------+
```
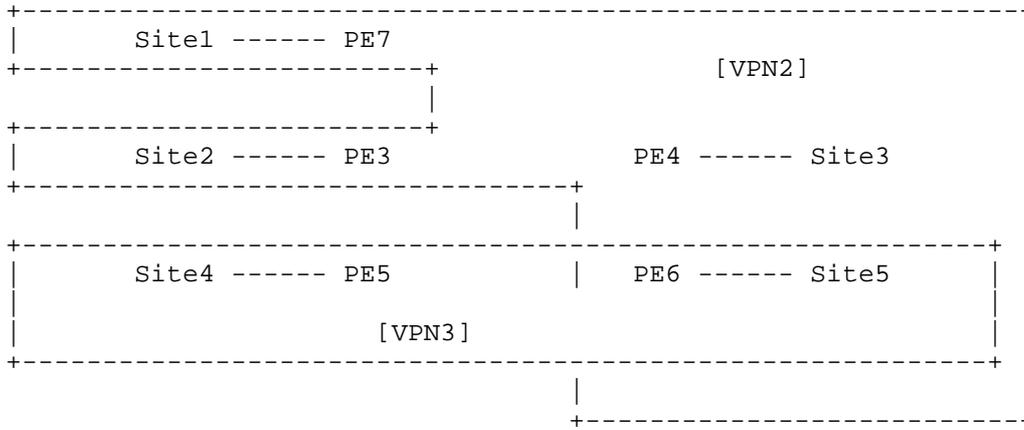
In the example above, Site5 is part of two VPNs: VPN3 and VPN2. It
will play a hub-role in VPN2 and an any-to-any role in VPN3. We can
express such multiVPN scenario as follows:

```
   <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
   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 (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 the customer constraints, if the
   constraint is too strict and can not be filled, the management system

MUST not provision the site and SHOULD provide an information to the
user. How the information is provided is out of scope of the
document. It would then be up to the user to relax the constraint or
not.

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

In addition to parameters and constraints, the management system
decision MAY be based on any other internal constraint that are up to
the service provider: least load, distance ...

6.6.1. Constraint: Device

In 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
he ordered.

```
     New York Site


     +-----------------+                Site
     | +-------------+ |--------------------------------
     | | Manhattan   | |
     | |         CE1********* (site-network-access#1) ******
     | +-------------+ |
     | +-------------+ |
     | | Brooklyn  CE2********* (site-network-access#2) ******
     | +-------------+ |
     |                 |--------------------------------
     +-----------------+
```
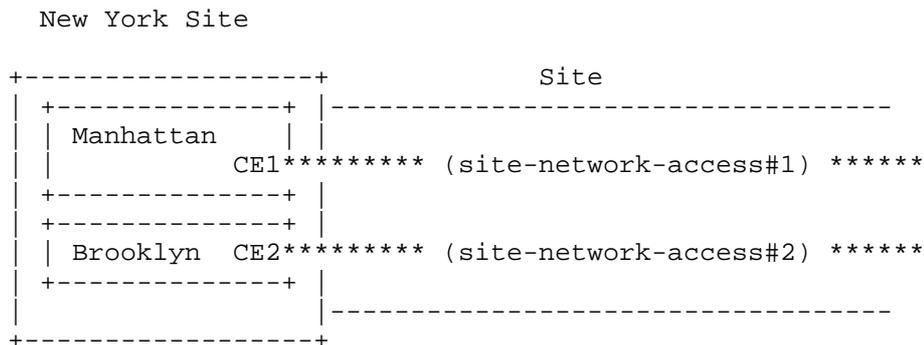
In the figure above, the site-network-access#1 is associated to CE1
in the service request. The service provider must ensure the
provisionning of this connection.

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
(service provider side). A particular location must be associated to
each site network access when configuring it. The service provider
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 case of 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 the customer ordered. As each device is already associated to a particular location, in such case the location information is retrieved from the device location. In case of 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 the placement.

```
                                    PoP#1 (New York)
                                  +---------+
                                  |   PE1   |
                  Site #1 ---...  |   PE2   |
                  (Atlantic City) |   PE3   |
                                  +---------+


                                    PoP#2 (Washington)
                                  +---------+
                                  |   PE4   |
                                  |   PE5   |
                                  |   PE6   |
                                  +---------+


                                    PoP#3 (Philadelphia)
                                  +---------+
                                  |   PE7   |
                  Site #2 CE#1---...  |   PE2   |
                  (Reston)         |   PE9   |
                                  +---------+
```
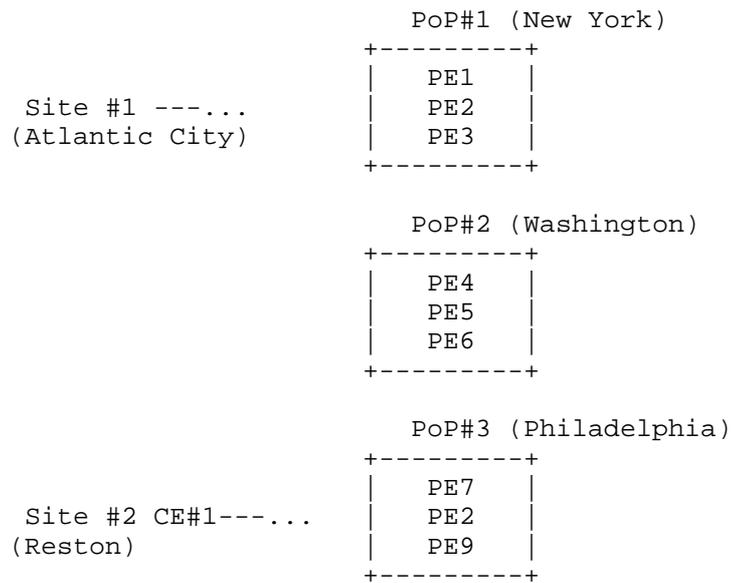
In the example above, Site#1 is a customer managed site with a location L1, while Site#2 is a provider-managed site for which a CE#1 was ordered, Site#2 is configured with L2 as location. When configuring a site-network-access for Site#1, the user will need to reference the location L1, so the management system will know that the access will need to terminate on this location. Then this management system may mesh Site#1 on a PE in the Philadelphia PoP for distance reason. It may also take into account resources available on PEs to determine the exact target PE (e.g. least loaded). Regarding Site#2, the user is expected to configure the site-network-access with a device-reference to CE#1, so the management system will know that the access must terminate on the location of CE#1 and must be connected to CE#1. For placing the service provider side of the access connection, in case of shortest distance 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 used as first input :

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

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

   o The "bearer-reference" is used in case the customer has already
      ordered a network connection to the service provider apart of the
      IPVPN site and wants to reuse this connection. The string used in
      an internal reference from the service provider describing 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 of the document but as a pure example,
      when the customer ordered the bearer (through a process out of
      this model), the service provider may had provided the bearer
      reference that can be used for provisionning services on top.

   Any other internal parameters from the service provider can be used
   in addition. The management system MAY use other parameters such as
   the requested svc-input-bandwidth and svc-output-bandwidth to help to
   decide the access type to be used.

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 no
   constraint but is expected allocation of a unique bearer per site-
   network-access.

   In order to help 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") or 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 are belonging 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: e.g. "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 or "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 the current site-
network-access is not belonging 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 or 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
   configuration of scenarios where a group of site-network-access has a
   constraint between each other. As an example if we want a set of
   sites (site#1 up to #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 be also
different than the one used by the current site-network-access. This
can be used to express that a group of site has some constraints
against another group of sites, but there is no constraint within the
group. As an example, if we consider a set of 6 sites with two sets
and we want to ensure that a site in the first set must be pop-
diverse from a site in the second set.

```
  <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. Impossible access placement

   Some impossible placement scenarios may be created through the
   proposed configuration framework. Impossible scenarios could come
   from too restrictive constraints leading to impossible placement in
   the network or conflicting constraints that would also lead to
   impossible placement. An example of conflicting rules would be to
   request a site-network-access#1 to be pe-diverse from a site-network-

   access#2 and to request at the same time that site-network-access#2
   to 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 and the customer wants the two
   site-network-accesses to be meshed on different PoPs for resiliency
   purpose.

```
                                       PoP#1
   +-------+                        +---------+
   |       |                        |   PE1   |
   |       |---site_network_access#1 ----    |   PE2   |
   |       |                        |   PE3   |
   |       |                        +---------+
   | Site#1|
   |       |                           PoP#2
   |       |                        +---------+
   |       |                        |   PE4   |
   |       |---site_network_access#2 ----   |   PE5   |
   |       |                        |   PE6   |
   |       |                        +---------+
   |       |
   +-------+
```

   This scenario can be expressed in the following way:

   <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:

```
      <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 to have all branch offices to be connected on the
   same PE.

   He wants to express that three branch offices cannot be connected on
   the same linecard. And the other branch offices must be connected on

   a different PoP. Those other branch offices cannot also be connected
   on the same linecard.

                                        PoP#1
                              +---------+
                              | PE1 |
                Office#1 ---... | PE2 |
                Office#2 ---... | PE3 |
                Office#3 ---... | PE4 |
                              +---------+

                                        PoP#2
                              +---------+
                Office#4 ---... | PE4 |
                Office#5 ---... | PE5 |
                Office#6 ---... | PE6 |
                              +---------+

   This scenario can be expressed in the following way:

   o We need to create two sets of sites: set#1 composed of Office#1 up
      to 3, set#2 composed of Office#4 up to 6.

   o Sites within set#1 must be pop-diverse from sites within set#2 and
      vice versa.

   o Sites within set#1 must be linecard-diverse from other sites in
      set#1 (same for set#2).

   <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>
        <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>
    <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>
         <constraint>
          <constraint-type>linecard-diverse</constraint-type>
          <target>
           <group>
            <group-id>10</group-id>
           </group>
          </target>
         </constraint>
```
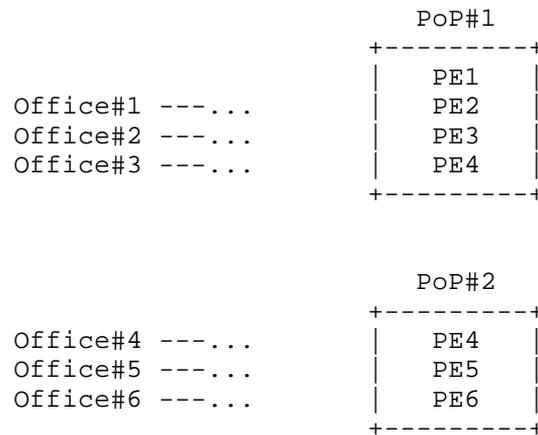
```
           </constraints>
          </access-diversity>
          <vpn-attachment>
           <vpn-id>VPNA</vpn-id>
           <site-role>spoke-role</site-role>
          </vpn-attachment>
         </site-network-access>
       </site>
       <site>
        <site-id>SITE3</site-id>
        <site-network-accesses>
         <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>SITE4</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>
   <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>
       <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>
   <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>
        <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 a cheaper cost, a customer wants to
order two parallel site-network-accesses that will be connected to
the same PE.

```
        *******SNA1**********
Site 1 *******SNA2**********  PE1
```

This scenario can be expressed in the following way:

```
<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 site which is dualhomed. The dualhoming must be done
   on two different PEs. The customer wants also to implement two
   subVPNs on those multihomed accesses.

```
   +-----------------+         Site                  +-----+
   |                 |------------------------------/   +----+
   |                 |****(site-network-access#1)******| VPNB /       \
   |  New York Office|****(site-network-access#2)************| VPN C  |
   |                 |                          +----\      /
   |                 |                              +-----+
   |                 |
   |                 |                              +------+
   |                 |                             /    +-----+
   |                 |****(site-network-access#3)******|  VPNB /       \
   |                 |****(site-network-access#4)************| VPN C  |
   |                 |                          +------\      /
   |                 |------------------------------         +---+
   +-----------------+
```

   This scenario can be expressed in the following way:

   o The site will have 4 site network accesses (2 subVPN coupled with
      dual homing).

   o Site-network-access#1 and #3 will correspond to the multihoming of
      the subVPN B. A PE-diverse constraint is required between them.

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

   o To ensure proper usage of the same bearer for the subVPN, site-
      network-access #1 and #2 must share the same bearer as site-
      network-access #3 and #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-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 is a critical parameter of PE-based L3VPNs as
   described in [RFC4364] that allows to distinguish common addressing
   plans in different VPNs. As for route-targets, it is expected that a
   management system will allocate a VRF on the target PE and a route-
   distinguisher for this VRF.

   If a VRF already exists on the target PE, and the VRF fulfils 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 fulfils the
   connectivity constraints for a site is out of scope of this document.

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

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

The allocation of route-distinguishers MAY be done in the same way as route-targets. The example provided in Section 6.2.1.1 could be reused.

Note that a service provider MAY configure a target PE for an automated allocation of route-distinguishers. In this case, there will be no need for any backend system to allocate a route-distinguisher value.

6.7. Site network access availability

A site may be multihomed, meaning it has multiple site-network-access points. Placement constraints defined in previous sections will help to 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 the site redundancy. The "access-priority" leaf defines a preference for a particular access. This preference is used to model loadbalancing or primary/backup scenarios. The higher the access-priority the higher the preference will be.

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

```
Hub#1 LAN (Primary/backup)        Hub#2 LAN (Loadsharing)
   |                                 |
   |      access-priority 1       access-priority 1    |
   |--- CE1 ------- PE1          PE3 --------- CE3 --- |
   |                                 |
   |                                 |
   |--- CE2 ------- PE2          PE4 --------- CE4 --- |
   |      access-priority 2       access-priority 1    |


                        PE5
                         |
                         |
                         |
                        CE5
                         |
                Spoke#1 site (Single-homed)
```
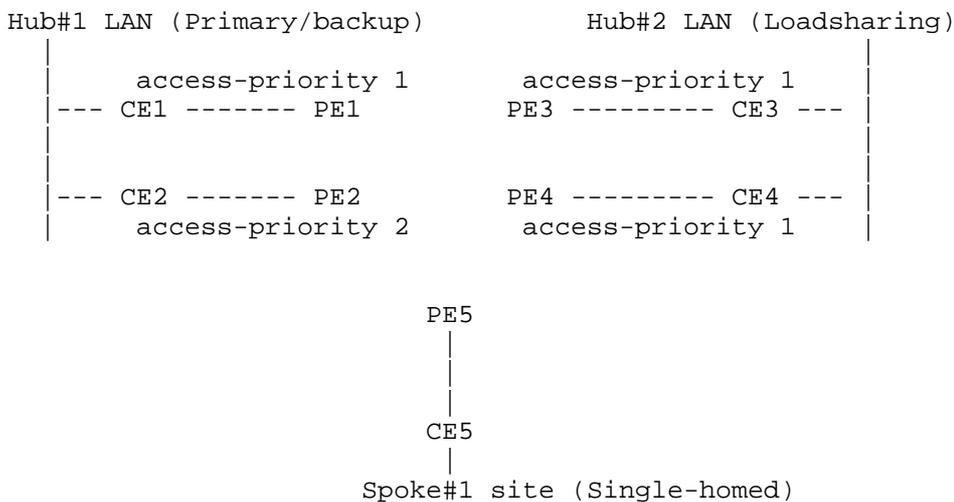
In the figure above, Hub#2 requires loadsharing so all the site-network-accesses must use the same access-priority value. On the

contrary, as Hub#1 requires primary/backup, an higher access-priority
will be configured on the primary access.

More complex scenarios can be modeled. Let's consider a Hub site
with five accesses to the network (A1,A2,A3,A4,A5). The customer
wants to loadshare its traffic on A1,A2 in the nominal situation. If
A1 and A2 fails, he wants to loadshare its traffic on A3 and A4, and
finally if A1 to A4 are down, he wants to use A5. We can model it
easily by configuring the following access-priorities: A1=100,
A2=100, A3=50, A4=50, A5=10.

The access-priority has some limitation. A scenario like the
previous one with five accesses but with the constraint of having
traffic loadshared between A3 and A4 in case of A1 OR A2 being down
is not achievable. But the authors consider that the access-priority
covers most of the deployment use cases and the model can still be
extended by augmentation to support additional use cases.

6.8. Traffic protection

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

```
     Site#1                          Site#2
CE1 ----- PE1 -- P1          P3 -- PE3 ---- CE3
 |                            |              |
 |                            |              |
CE2 ----- PE2 -- P2          P4 -- PE4 ---- CE4
        /
       /
CE5 ----+
   Site#3
```
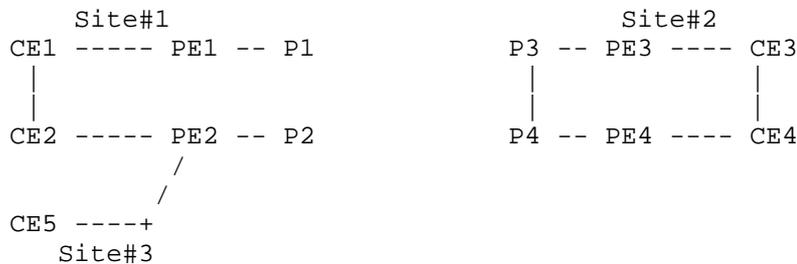
In the figure above, we consider an IPVPN service with three sites
including two dual homed sites (site#1 and #2). For dual homed
sites, we consider PE1-CE1 and PE3-CE3 as primary, and
PE2-CE2,PE4-CE4 as backup for the example (even if protection also
applies to loadsharing scenarios).

In order to protect Site#2 against a failure, user may set the
"traffic-protection/enabled" leaf to true for site#2. How the

traffic protection will be implemented is out of scope of the
document. But as an example, in such a case, if we consider traffic
coming from a remote site (site#1 or site#3), where the primary path
is to use PE3 as egress PE. PE3 may have preprogrammed a backup
forwarding entry pointing to backup path (through PE4-CE4) for all
prefixes going through PE3-CE3 link. How the backup path is computed
is out of scope of the document. When PE3-CE3 link fails, traffic is
still received by PE3 but PE3 switch automatically traffic to the
backup entry, path will so be PE1-P1-(...)-P3-PE3-PE4-CE4 until
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 by 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

A traffic encryption can be requested on the connection. It may be
performed at layer 2 or layer 3 by selecting the appropriate
enumeration in "layer" leaf. For example, a service provider may use
IPSec when a customer is requesting layer 3 encryption. The
encryption profile can be a service provider defined profile or a
customer specific.

When a service provider profile is used and a key (e.g. a preshared
key) is allocated by the provider to be used by a customer, the
service provider 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 preshared
key for authentication with the preshared key provided through the
NETCONF or RESTCONF request. A secure channel must be used to ensure
that the preshared key cannot be intercepted.

It may be necessary for the customer to change the preshared key on a
regular basis for security reasons. To perform a key change, the
user can request to the service provider by submitting a new
preshared key for the site configuration (as displayed below). This
mechanism may not to 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>
```

   An hitless key change mechanism may be added through augmentation.

   Other key management methodology may be added through augmentation.
   A "pki" empty container has been created to help support of PKI
   through augmentation.

6.10. Management

   The model proposes three types of common management options:

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

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

   o co-managed: the CE router is primarly managed by the provider and
      in addition SP lets customer accessing the CE for some
      configuration/monitoring purpose. In the co-managed mode the
      responsibility boundary is the same as the provider-managed model.

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

   In case of "co-managed", 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: bgp, rip, ospf, static, direct, vrrp.

   The routing protocol defined applies at the provider to customer
   boundary. Depending on the management of the management model, it
   may apply to the PE-CE boundary or CE to customer boundary. In 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 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, it is expected that the PE-
   CE routing will be configured based on the service provider 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 customer managed site case.

6.11.1. Dual stack handling

   All routing protocol types support dual stack by using 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. Direct LAN connection onto SP network

   Routing-protocol "direct" SHOULD be used when a customer LAN is
   directly connected to the provider network and must be advertised in
   the IPVPN.

        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. Direct LAN connection onto SP network with redundancy

   Routing-protocol "vrrp" SHOULD be used when a customer LAN is
   directly connected to the provider network and must be advertised in
   the IPVPN and LAN redundancy is expected.

        LAN attached directly to provider network with LAN redundancy:

        192.0.2.0/24 ------ PE1
                       |
                       +--- PE2

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

6.11.4. Static routing

   Routing-protocol "static" MAY be used when a customer LAN is
   connected to the provider network through a CE router and must be
   advertised in the IPVPN.

```
                                    Static rtg
            192.0.2.0/24 ------ CE -------------- PE
                              |                   |
                              |       Static route 192.0.2.0/24 nh CE
                     Static route 0.0.0.0/0 nh PE
```

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

6.11.5. RIP routing

   Routing-protocol "rip" MAY be used when a customer LAN is connected
   to the provider network through a CE router and must be advertised in
   the IPVPN. For IPv4, the model assumes usage of RIP version 2.

   In case of dual stack routing requested through this model, the
   management system will be responsible to configure rip (including
   right version number) and associated address-families on network
   elements.

```
                                    RIP rtg
            192.0.2.0/24 ------ CE -------------- PE
```

6.11.6. OSPF routing

   Routing-protocol "ospf" MAY be used when a customer LAN is connected
   to the provider network through a CE router and must be advertised in
   the IPVPN.

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

```
                          +--------------------+
                          |                    |
                 OSPF     |                    | OSPF
                 area 1   |                    | area 2
        (OSPF             |                    |          (OSPF
         area 1) --- CE ---------- PE          PE ----- CE --- area 2)
                          |                    |
                          +-------------------+
```

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 to check if
there is already a shamlink configured for this VPN and area between
the same pair of PEs. If there is no existing shamlink, the
management system will provision it. This shamlink MAY be reused by
other sites.

```
                     +----------------------+
                     |                      |
                     |                      |
                     |  PE (--shamlink--)PE |
                     |  |                |  |
                     +----|---------------|--+
                          | OSPF area1    | OSPF area 1
                          |               |
                       CE1               CE2
                          |               |
                     (OSPF area1)    (OSPF area1)
                          |               |
                       +---------------+
```
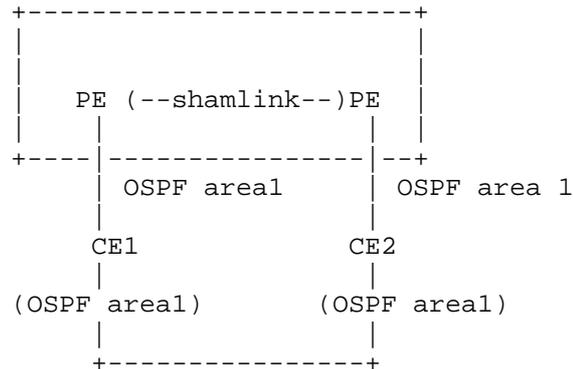
Regarding Dual stack support, user MAY specify both IPv4 and IPv6
address families, if both protocols should be routed through OSPF.
As OSPF uses separate protocol instances for IPv4 and IPv6, the
management system will need to configure both ospf version 2 and
version 3 on the PE-CE link.

Example of OSPF routing parameters in 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 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

   Routing-protocol "bgp" MAY be used when a customer LAN is connected
   to the provider network through a CE router and must be advertised in
   the IPVPN.

                                 BGP rtg
         192.0.2.0/24 ------ CE -------------- PE

   The session addressing will be derived from connection parameters as
   well as internal knowledge of SP.

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

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

   BGP activation requires SP to know the address of the customer peer.
   "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>

   This service configuration can be derived by management system into
   multiple flavors depending on SP flavor.

Example #1 of PE configuration done by management system
(single session 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 #2 of PE configuration done
by 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 #3 of PE configuration done
by management system (multisession):

```
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 PE
and CE (WAN link bandwidth). The requested bandwidth is expressed as
svc-input-bandwidth and svc-output-bandwidth in bits per seconds.
Input/output direction is using customer site as 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 site-network-access
level.

Using a different input and output bandwidth will allow service
provider to know if customer allows for asymmetric bandwidth access
like ADSL. It can also be used to set rate-limit in a different way
upload and download on a symmetric bandwidth access.

The bandwidth is a service bandwidth: expressed primarily as IP
bandwidth but if the customer enables MPLS for carrier's carrier,
this becomes MPLS bandwidth.

6.12.2. QoS

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

o qos-classification-policy: define a set of ordered rules to
   classify customer traffic.

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

6.12.2.1. QoS classification

QoS classification rules are handled by qos-classification-policy.
The qos-classification-policy 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 more specific flow definition (based layer
3 source and destination address, layer 4 ports, layer 4 protocol).
When a flow definition is used, the user can use 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 of this
document and an implementation may not support this criterion and
should advertise a deviation in this case. A rule that does not have
a match statement is considered as a match-all rule. A service
provider may implement a default terminal classification rule if the
customer does not provide it. It will be up to the service provider
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 service provider, so it will be necessary for the service
provider to advise customer on usage of application matching.

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

```
                                  Provider network
                        +---------------------+
     192.0.2.0/24
   198.51.100.0/24 ---- CE --------- PE

    Traffic flow
    ---------->
```
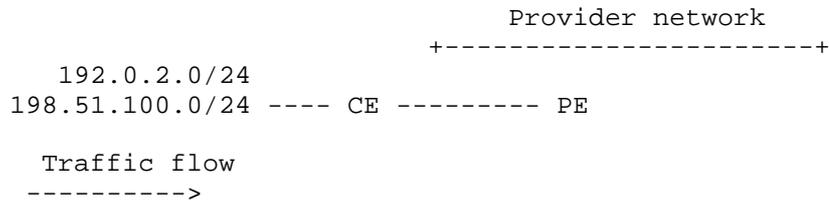
   In the figure above, the management system should implement the
   classification rule:

   o in the ingress direction on the PE interface, if the CE is
      customer managed.

   o in the ingress direction on the CE interface connected to 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:

o HTTP traffic from 192.0.2.0/24 LAN destinated to 203.0.113.1/32
   will be classified in DATA2.

o FTP traffic from 192.0.2.0/24 LAN destinated to 203.0.113.1/32
   will be classified in DATA2.

o Peer to peer traffic will be classified in DATA3.

o All other traffic will be classified in DATA1.

The order of rules is really 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 rule is defined by the "id" leaf. The
lowest "id" MUST be processed first.

6.12.2.2. QoS profile

User can choose between standard profile provided by the operator or
custom profile. The qos-profile defines the traffic scheduling
policy to be used by the service provider.

```
                              Provider network
                      +----------------------+
192.0.2.0/24
198.51.100.0/24 ---- CE --------- PE
                       \        /
                      qos-profile
```

In case of 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 example
of implementation, a device scheduling policy may be implemented both
at PE and CE side on the WAN link. In case of customer managed
connection, the provider is only responsible to ensure scheduling
from SP network to the customer site. As example of implementation,
a device scheduling policy may be implemented only at PE side on the
WAN link towards the customer.

A custom qos-profile is defined as a list of class of services and
associated properties. The properties are:

o 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 is implemented at CE side the svc-output-bandwidth
   is taken into account as reference. When it is implemented at PE
   side, the svc-input-bandwidth is used.

o 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 service provider
   implementation: a strict priority queueing may be used on the
   access and in the core network, and/or a low latency routing may
   be created for this traffic class.

o 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 service provider
   implementation: a strict priority queueing may be used on the
   access and in the core network, and/or a jitter-aware routing may
   be created for this traffic class.

o bandwidth: used to define a guaranteed amount of bandwidth for the
   class of service. It is expressed as a percentage. The
   guaranteed-bw-percent uses available bandwidth as a reference.
   When the qos-profile is implemented at CE side the svc-output-
   bandwidth is taken into account as reference. When it is
   implemented at PE side, the 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 the MPLS transport
   network.

Some constraints may not be offered by a service provider, in this
case a deviation should be advertised. In addition, due to the
network conditions, some constraints may not be completely fulfilled
by the service provider, in this case, the service provider should
advise the customer about the limitations. How this communication is
done is out of scope of 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
   lowest possible latency constraint. It defines also 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 100Mbps
= 10Mbps). In case of congestion, the REAL_TIME traffic can go up to
10Mbps (let's assume that only 5Mbps are consumed). DATA1 and DATA2
will share the remaining bandwidth (95Mbps) according to their
percentage. So the DATA1 class will be served with at least 76Mbps
of bandwidth while the DATA2 class will be served with at least
4.75Mbps. The latency boundary information of the data class may
help the service provider to 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 service
provider.

6.12.3. Multicast

   The multicast section defines the type of site in the customer
   multicast service topology: source, receiver, or both. These
   parameters will help management system to optimize the multicast
   service. User can also define the type of multicast relation with
   the customer: router (requires a protocol like PIM), host (IGMP or
   MLD), or both. Address family (IPv4 or IPv6 or both) can also be
   defined.

6.13. Enhanced VPN features

6.13.1. Carrier's Carrier

   In case of Carrier's Carrier ([RFC4364]), a customer may want to
   build MPLS service using an IPVPN to carry its traffic.

```
            LAN customer1
                 |
                 |
              CE1
                 |
                 | ------------
            (vrf_cust1)
             CE1_ISP1
                 |              ISP1 PoP
                 | MPLS link
                 | ------------
                 |
            (vrf ISP1)
               PE1


             (...)                Provider backbone


             PE2
            (vrf ISP1)
                 |
                 | -----------
                 |
                 | MPLS link
                 |              ISP1 PoP
            CE2_ISP1
            (vrf_cust1)
               |------------
                 |
              CE2
                 |
            Lan customer1
```
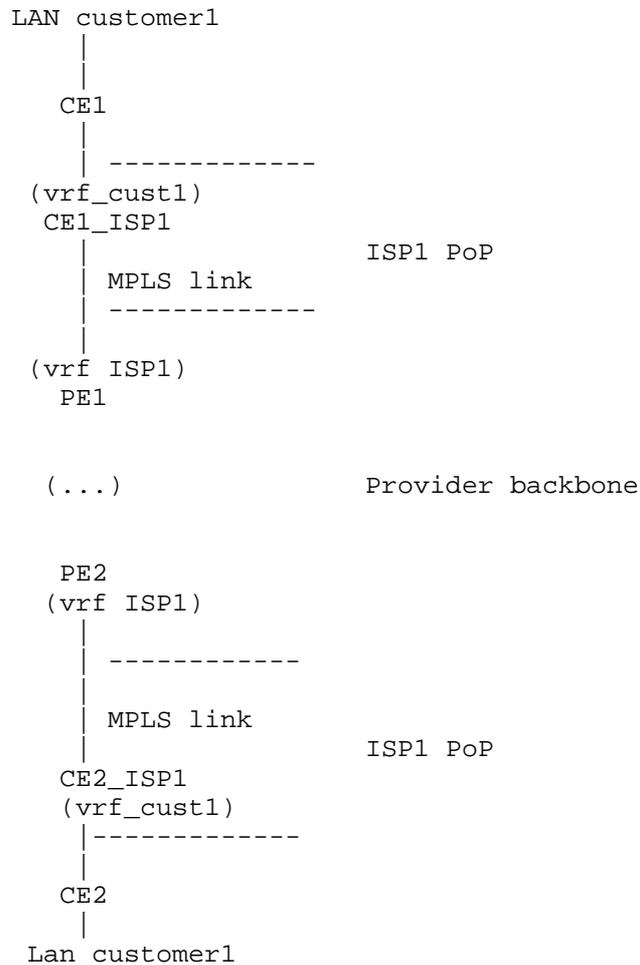
   In the figure above, ISP1 resells IPVPN service but has no core
   network infrastructure between its PoPs. ISP1 uses an IPVPN as core
   network infrastructure (belonging to another provider) between its
   PoPs.

   In order to support CsC, the VPN service must be declared MPLS
   support using the "carrierscarrier" leaf set to true in vpn-service.
   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 case of LDP, an IGP routing protocol MUST also be

activated. In case of BGP signalling, BGP MUST also be configured as
routing-protocol.

In case Carrier's Carrier is enabled, the requested svc-mtu 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. In case, a customer is using directly this
service model as an API (through REST or NETCONF for example) to
order a particular service, the service provider should provide a
list of authorized identifiers. In case of cloud-access, the service
provider will provide the identifiers associated of each available
CSP. The same applies to other identifiers like std-qos-profile, oam
profile-name, provider-profile for encryption ...

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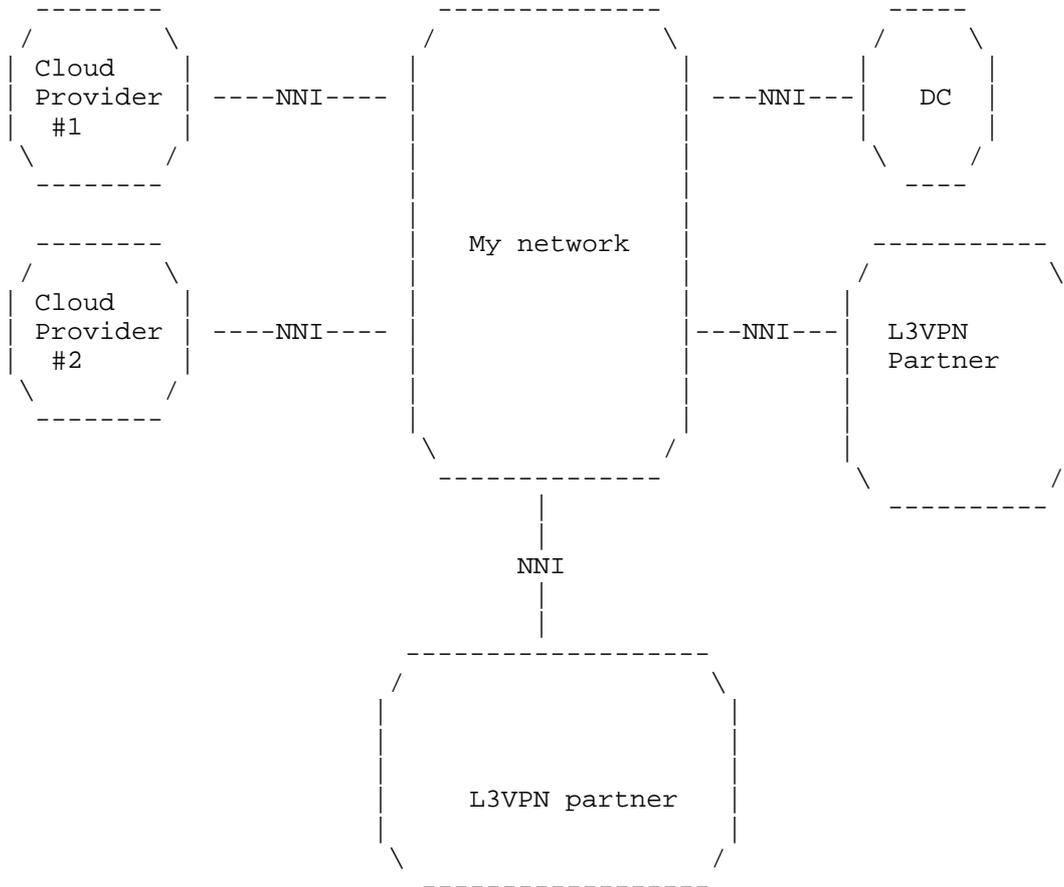of scope of this document.

6.15. Defining NNIs

An autonomous system 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 across different autonomous systems in different geographic
areas or across different service providers. The connection between
autonomous systems is established by the Service Providers and is
seamless to the customer.

Some examples are: Partnership between service providers (carrier,
cloud ...) to extend their VPN service seamlessly, or internal
administrative boundary within a single service provider (Backhaul vs
Core vs Datacenter ...).

NNIs (Network to Network Interfaces) have to be defined to extend the
VPNs across multiple autonomous systems.

[RFC4364] defines multiple flavors of VPN NNI implementations. Each
implementation has different pros/cons that are outside the scope of
this document. As an example: In an Inter-AS Option A, ASBR peers
are connected by multiple interfaces with at least one interface
which VPN spans the two autonomous systems. These ASBRs associate
each interface with a VPN routing and forwarding (VRF) instance and a
Border Gateway Protocol (BGP) session to signal unlabeled IP
prefixes. 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).

```
    --------              -------------            -----
   /        \            /             \          /     \
  | Cloud    |          |               |        |       |
  | Provider | ----NNI---- |            |  ---NNI---| DC  |
  |   #1     |          |               |        |       |
   \        /           |               |         \     /
    --------            |               |          ----
                        |               |
    --------            |  My network   |         ----------
   /        \           |               |        /          \
  | Cloud    |          |               |       |            |
  | Provider | ----NNI---- |            | ---NNI---| L3VPN    |
  |   #2     |          |               |       | Partner    |
   \        /           |               |       |            |
    --------            |               |       |            |
                         \             /        |            |
                          -------------          \          /
                               |                  ----------
                               |
                              NNI
                               |
                               |
                     ------------------
                    /                  \
                   |                    |
                   |                    |
                   |                    |
                   |   L3VPN partner    |
                   |                    |
                    \                  /
                     ------------------
```

The figure above describes a service provider network "My network"
that has several NNIs. This network uses NNI to:

o increase its footprint by relying on L3VPN partners.

o connect its own datacenter services to the customer IPVPN.

o enable customer to access to its private resources located in
   private cloud owned by some cloud service providers.

6.15.1. Defining NNI with option A flavor

```
            AS A                                        AS B
     --------------------                        -------------------
    /                    \                      /                    \
   |                     |   |                  |  |                   |
   |           ++++++++ InterAS link +++++++    |                     |
   |           +       +_____ +      + |                     |
   |           +  (VRF1)--(VPN1)----(VRF1)  +    |                     |
   |           + ASBR +             + ASBR +     |                     |
   |           +  (VRF2)--(VPN2)----(VRF2)  +    |                     |
   |           +       +_____+     +    |                     |
   |           ++++++++              ++++++++     |                     |
   |              |                     |         |                     |
   |              |                     |         |                     |
   |              |                     |         |                     |
   |           ++++++++ InterAS link +++++++      |                     |
   |           +       +_____ +      +   |                     |
   |           +  (VRF1)--(VPN1)----(VRF1)  +      |                     |
   |           + ASBR +             + ASBR +       |                     |
   |           +  (VRF2)--(VPN2)----(VRF2)  +      |                     |
   |           +       +_____+     +      |                     |
   |           ++++++++              ++++++++        |                     |
   |              |                     |            |                     |
   |              |                     |            |                     |
    \             /                      \                    /
     --------------------                  -------------------
```

   In option A, the two ASes are connected between each other with
   physical links on Autonomous System Border Routers (ASBR). There may
   be multiple physical connections between the ASes for a resiliency
   purpose. A VPN connection, physical or logical (on top of physical),
   is created for each VPN that needs to cross the AS boundary. A back-
   to-back VRF model is so created.

   This VPN connection can be seen as a site from a service model
   perspective. Let's say that AS B wants to extend some VPN connection
   for VPN C on AS A. Administrator of AS B can use this service model
   to order a site on AS A. All connection scenarios could be realized
   using the current model features. As an example, the figure above,
   where two physical connections are involved with logical connections
   per VPN on top, could be seen as a dualhomed subVPN scenario. And
   for example, administrator from AS B will be able to choose the
   appropriate routing protocol (e.g. ebgp) to dynamically exchange
   routes between ASes.

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

Example: a customer wants from its cloud service provider A to attach
its virtual network N to an existing IPVPN (VPN1) he has from a L3VPN
service provider B.

```
     CSP A                                    L3VPN SP B

    ----------------              -------------------
   /                \            /                   \
  |      |          |          |                      |
  |  VM --|       +++++++  NNI    +++++++            |---- VPN1
  |      |       +      +_____+      +           |     Site#1
  |      |-------(VRF1)---(VPN1)--(VRF1)+            |
  |      |       + ASBR +         + ASBR +           |
  |      |       +      +_____+      +           |
  |      |       +++++++           +++++++           |
  |  VM --|          |               |               |---- VPN1
  |      |Virtual    |               |               |     Site#2
  |      |Network    |               |               |
  |  VM --|          |               |               |---- VPN1
  |      |           |               |               |     Site#3
   \                /            \                   /
    ----------------              -------------------
                                          |
                                          |
                                         VPN1
                                        Site#4
```

The cloud service provider or the customer may use our L3VPN service
model exposed by service provider B to create the VPN connectivity.
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 a new site creation with a single site-
network-access (single homing used in the diagram). As 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 describes what could be the 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 the cloud-access container
usage as the cloud-access provides a public cloud access while this

example enables access to private resources located in a cloud
service provider network.

6.15.2. Defining NNI with option B flavor

```
           AS A                                      AS B
     --------------------               --------------------
    /                    \             /                    \
    |                    |             |                    |
    |          ++++++++ InterAS link +++++++              |
    |          +      +_____  +      +           |
    |          +      +                +      +           |
    |          + ASBR +<---mpebgp--->+ ASBR +             |
    |          +      +                +      +           |
    |          +      +_____+      +             |
    |          ++++++++                ++++++++           |
    |             |                       |               |
    |             |                       |               |
    |             |                       |               |
    |          ++++++++ InterAS link +++++++              |
    |          +      +_____  +      +           |
    |          +      +                +      +           |
    |          + ASBR +<---mpebgp--->+ ASBR +             |
    |          +      +                +      +           |
    |          +      +_____+      +             |
    |          ++++++++                ++++++++           |
    |             |                       |               |
    |             |                       |               |
    |             |                       |               |
    \             /             \                    /
     --------------------               --------------------
```

   In option B, the two ASes are connected between each other with
   physical links on Autonomous System Border Routers (ASBR). There may
   be multiple physical connections between the ASes for a resiliency
   purpose. The VPN "connection" between ASes is done by exchanging VPN
   routes through MP-BGP.

   There are multiple flavors of implementations of such NNI, for
   example:

   1. The NNI is a provider internal NNI between a backbone and a DC.
       There is enough trust between the domains to not filter the VPN
       routes. So all the VPN routes are exchanged. Route target
       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 route-targets only. Each provider is
    authorized to use the route-target values from the other
    provider.

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

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

Case 2 requires to maintain some route-target filtering policy on
ASBRs. From a service modeling point of view, it is necessary to
agree on the list of route target to authorize.

In case 3, both ASes need to agree on the VPN route-target to
exchange and in addition how to map a VPN route-target from AS A to
the corresponding route-target in AS B (and vice-versa).

Those modelings are currently out of scope of this document.

```
       Cloud SP                          L3VPN SP B
          A
     ----------------           ------------------
    /            \  |          /                \
   |      |       |          |
   | VM --|   +++++++   NNI   +++++++ |---- VPN1
   |      |   +       +_____+      +  |      Site#1
   |      |-------+      +      +      +  |
   |      |   + ASBR +<-mpebgp->+ ASBR +  |
   |      |   +       +_____+      +  |
   |      |   +++++++         +++++++ |
   | VM --|      |          |          |---- VPN1
   |      |Virtual   |          |          |      Site#2
   |      |Network   |          |          |
   | VM --|      |          |          |---- VPN1
   |      |      /          |          |      Site#3
    \ ---------------- /          |          |
     ----------------           |          |
                              \       ------------------  /
                                ------------------
                                    |
                                    |
                                  VPN1
                                  Site#4
```
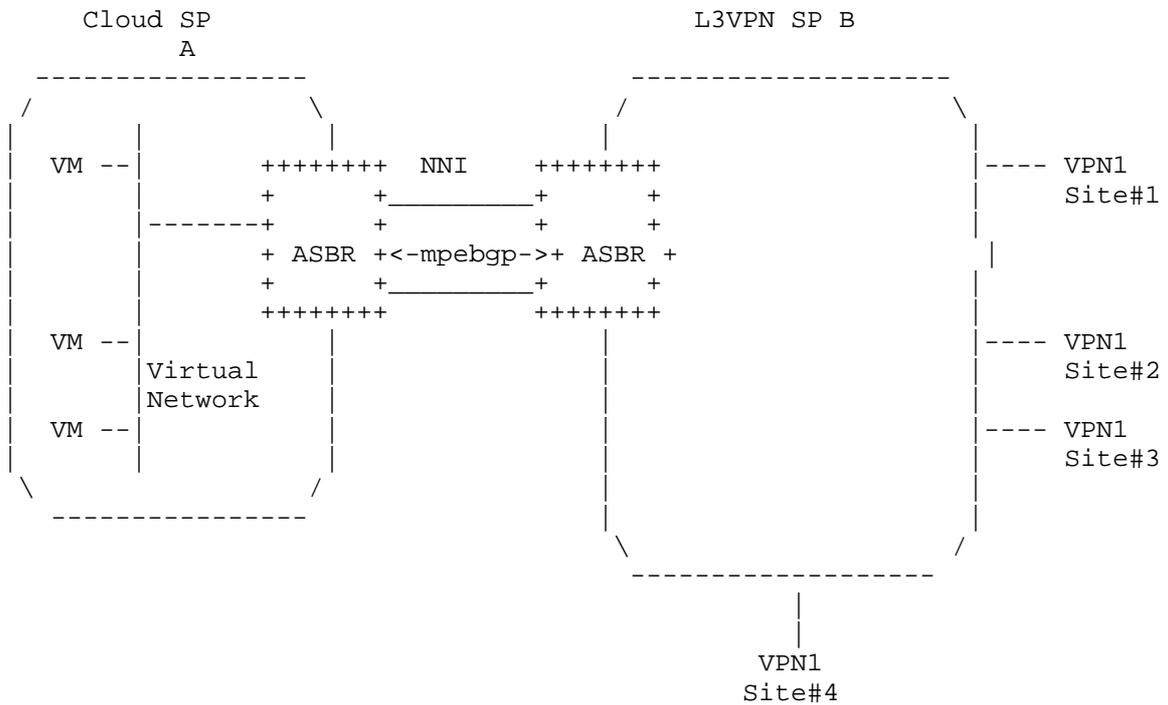
   The example above describes an NNI connection between the service
   provider network B and a cloud service provider A. Both service
   providers do not trust themselves and use a different route-target
   allocation policy. So, in term of implementation, the customer VPN
   has a different route-target in each network (RT A in CSP A and RT B
   is CSP B). In order to connect the customer virtual network in CSP A
   to the customer IPVPN (VPN1) in SP B network, CSP A should request SP
   B to open the customer VPN on the NNI (accept the appropriate RT).
   Who does the RT translation is up to an agreement between the two
   service providers: SP B may permit CSP A to request VPN (RT)
   translation.

6.15.3. Defining NNI with option C flavor

```
              AS A                                    AS B
    --------------------                    --------------------
   /                    \                  /                    \
   |         |         |                  |         |         |
   |         |         |                  |         |         |
   |         |  ++++++++ Multihop ebgp++++++++      |         |
   |         |  +       +            +       +      |         |
   |         |  +       +            +       +      |         |
   |         |  + RGW   +<---mpebgp--->+ RGW  +      |         |
   |         |  +       +            +       +      |         |
   |         |  +       +            +       +      |         |
   |         |  ++++++++            ++++++++      |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   |         |  ++++++++ InterAS link ++++++++      |         |
   |         |  +       +_____ +       +      |         |
   |         |  +       +            +       +      |         |
   |         |  + ASBR  +            + ASBR  +      |         |
   |         |  +       +            +       +      |         |
   |         |  +       +_____+       +      |         |
   |         |  ++++++++            ++++++++      |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   |         |  ++++++++ InterAS link ++++++++      |         |
   |         |  +       +_____ +       +      |         |
   |         |  +       +            +       +      |         |
   |         |  + ASBR  +            + ASBR  +      |         |
   |         |  +       +            +       +      |         |
   |         |  +       +_____+       +      |         |
   |         |  ++++++++            ++++++++      |         |
   |         |      |                    |          |         |
   |         |      |                    |          |         |
   \                    /                  \                    /
    --------------------                    --------------------
```

   From a VPN service perspective, 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 and control plane
   are on different nodes, so the MP-BGP is multihop between routing
   gateway (RGW) nodes.

Modeling option B and C will be identical from a VPN service point of view.

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 on how a management system can use this model to configure an IPVPN service on network elements.

The example wants to achieve the provisionning of a VPN service for 3 sites using Hub and Spoke VPN service topology. One of the sites will be dual homed and loadsharing is expected.

```
+--------------------------------------------------------------+
|   Hub_Site  ------ PE1                 PE2 ------ Spoke_Site1 |
|      |                    +---------------------------------+
|      |                    |
|      |                    +---------------------------------+
|   Hub_Site  ------ PE3                 PE4 ------ Spoke_Site2 |
+--------------------------------------------------------------+
```
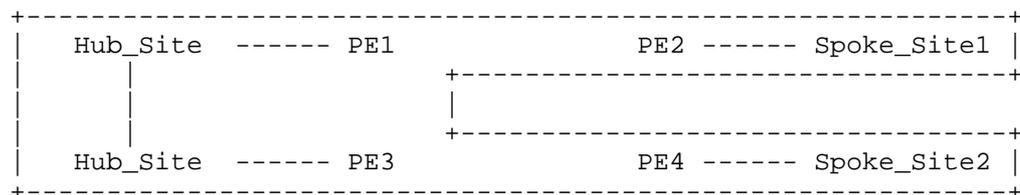
The following XML describes the overall simplified service configuration of this VPN.

```
<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) allocates VPN route-targets. 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 Spoke1.

```
<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 Spoke1 site, the
management system MUST allocate network resources for this site. It
MUST first determine the target network elements to provision the
access, and especially the PE router (and may be 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
Spoke1 requires PE diversity with Hub and that management system
allocate PEs based on lowest distance. Based on the location
information, the management system finds the available PEs in the
nearest area of 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
PE available pool. The management system can start provisioning the
PE node by using any mean (Netconf, CLI, ...). The management system
will check if a VRF is already present that fits the needs. If not,
it will provision the VRF: Route distinguisher will come from
internal allocation policy model, route-targets are coming from the
vpn-policy configuration of the site (management system allocated
some RTs for the VPN). As the site is a Spoke site (site-role), the
management system knows which RT must be imported and exported. As
the site is provider managed, some management route-targets 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
!
```

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, another
will be used for the CE configuration. Routing protocols will also
be configured between PE and CE and due to provider managed model,
the choice is up to service provider: BGP was chosen for the example.
This choice is independant of the routing protocol chosen by
customer. For the CE - LAN part, BGP will be used as requested in
the service model. Peering addresses will be derived from those of
the connection. As CE is provider managed, CE AS number can be
automatically allocated by the management system. Some provider
standard configuration templates 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 uploaded
    to the node by manual operation before sending the CE to customer
    premise. The CE configuration will be built as for the PE. Based on
    the CE type (vendor/model) allocated to the customer and 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 service provider OSS as both
    resources are managed internally. CE to LAN interface parameters
    like IP addressing are derived from ip-connection taking into account
    how management system distributes addresses between PE and CE within
    the subnet. This will allow to produce a plug'n'play configuration
    for the CE.

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
                                               ip-connection
 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
                                                  ip-connection
 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
                                                  ip-connection
 !
route-map STATIC2BGP permit 10
 match tag 10
 !
```

8. Interaction with Other YANG Modules

   As expressed in Section 5, this service module is intended to be
   instantiated in management system and not directly on network
   elements.

   It will be the role of the management system 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.

```
        L3VPN-SVC              |
          service model        |
                               |
             +---------------------+
             |   Service component | service datastore
             +---------------------+
                        |
                        |
             +---------------------+
          +----|   Config component   |-------+
         /     +---------------------+          \   Network
        /         /              \               \   Configuration
       /         /                \               \   models
      /         /                  \               \
   +++++++    ++++++++           ++++++++        +++++++
   + CEA + ------- + PE A +       + PE B + ----- + CEB + Config
   +++++++    ++++++++           ++++++++        +++++++ datastore

   Site A                                        Site B
```
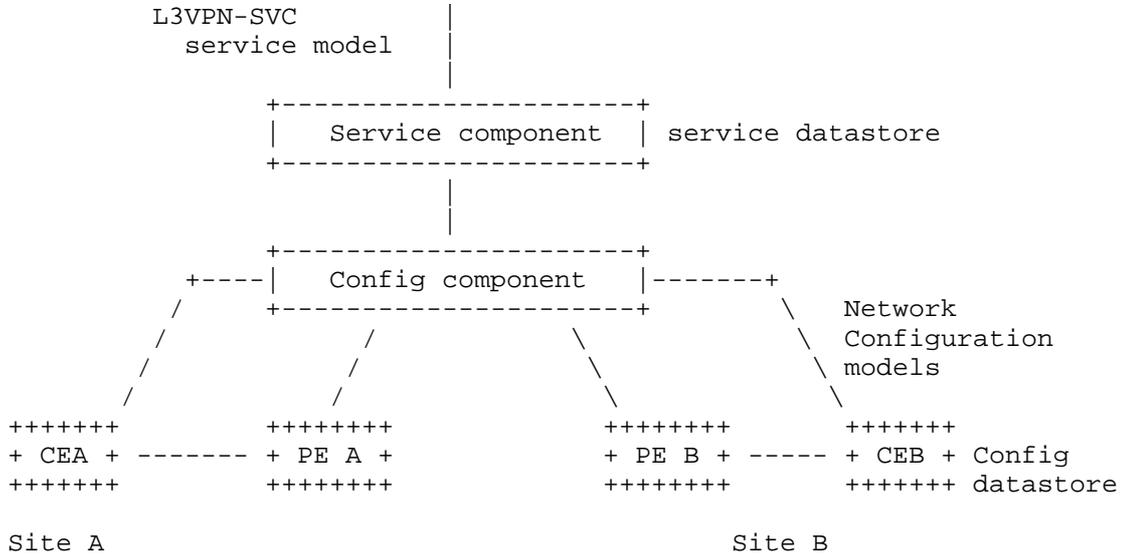
In the previous sections, we provided some example of translation of
service provisioning request to router configuration lines as an
illustration. In the NETCONF/YANG ecosystem, it will be expected
NETCONF/YANG to be used between configuration component and network
elements to configure the requested service on these elements.

In this framework, it is expected from standardization to also work
on specific configuration YANG modelization of service components on
network elements. There will be a strong relation 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.

Authors of this document are expecting definition of YANG models for
network elements on this non exhaustive list of items:

o VRF definition including VPN policy expression.

o Physical interface.

o IP layer (IPv4, IPv6).

o QoS: classification, profiles...

   o Routing protocols: support of configuration of all protocols
      listed in the document, as well as routing policies associated
      with these protocols.

   o Multicast VPN.

   o Network Address Translation.

   o ...

   Example of VPN site request at 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, it is expected that the service
component requests to the configuration component of the management
system the configuration of the service elements. If we consider
that service component selected a PE (PE A) as 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 may look like this:

```
     <if:interfaces>
         <if:interface>
          <if:name>eth0</if:name>
          <if:type>ianaift:ethernetCsmacd</if:type>
          <if:description>
           Link to CEA.
          </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:vrf</rt:type>
          <rt:description>VRF for CustomerA</rt:description>
          <l3vpn:route-distinguisher>
          100:1546542343
          </l3vpn:route-distinguisher>
```

```
        <l3vpn:import-rt>100:1</l3vpn:import-rt>
        <l3vpn:export-rt>100:1</l3vpn: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@2016-11-04.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: &lt;mailto:l3sm@ietf.org&gt;

     Editor:
      L3SM WG

     Chairs:
      Adrian Farrel, Qin Wu
      ";

    description
     "The YANG module defines a generic service configuration
     model for Layer 3 VPN common across all of the vendor
     implementations.";

    revision 2016-11-03 {
     description
      "Initial document";
     reference
       "RFC XXXX";
    }


    /* Features */

    feature cloud-access {
     description
      "Allow VPN to connect to a Cloud Service
      provider.";
    }
    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 carrier's carrier";
```

```
    }
    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 Class of Services";
    }
    feature qos-custom {
     description
     "Enables support of custom qos profile";
    }
    feature rtg-bgp {
     description
     "Enables support of BGP routing protocol.";
    }
    feature rtg-rip {
     description
     "Enables support of RIP routing protocol.";
    }
    feature rtg-ospf {
     description
     "Enables support of OSPF routing protocol.";
    }
    feature rtg-ospf-sham-link {
     description
     "Enables support of OSPF sham-links.";
    }
    feature rtg-vrrp {
     description
     "Enables support of 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 for always-on access
     constraint.";
    }
    feature requested-type {
     description
     "Enables support for requested-type access
     constraint.";
    }
    feature bearer-reference {
     description
     "Enables support for bearer-reference access
     constraint.";
    }

    /* Typedefs */

    typedef svc-id {
     type string;
     description
       "Defining a type of service component
       identificators.";
    }

    typedef template-id {
     type string;
     description
       "Defining a type of service template
       identificators.";
    }

    typedef address-family {
     type enumeration {
      enum ipv4 {
       description
         "IPv4 address family";
      }
      enum ipv6 {
       description
         "IPv6 address family";
      }
     }
     description
     "Defining a type for 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 applications";
   }
   identity file-transfer {
    base customer-application;
    description
      "Identity for file transfer applications (
      e.g. FTP, SFTP, ...)";
   }
   identity database {
    base customer-application;
    description
      "Identity for database applications";
   }
   identity social {
    base customer-application;
    description
      "Identity for social network applications";
   }
   identity games {
    base customer-application;
    description
      "Identity for gaming applications";
   }
   identity p2p {
    base customer-application;
    description
      "Identity for peer to peer applications";
```

```
      }
      identity network-management {
       base customer-application;
       description
        "Identity for management applications (e.g. telnet
        syslog, snmp ...)";
      }
      identity voice {
       base customer-application;
       description
        "Identity for voice applications";
      }
      identity video {
       base customer-application;
       description
        "Identity for video conference applications";
      }


      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 of the 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 a NNI option A connection.";
   }
   identity management {
    description
      "Base identity for site management scheme.";
   }
   identity co-managed {
    base management;
    description
      "Base identity for comanaged 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 a any to any IPVPN.";
    }
    identity spoke-role {
     base site-role;
     description
      "Spoke Site in a Hub & Spoke IPVPN.";
    }
    identity hub-role {
     base site-role;
     description
      "Hub Site in a Hub & Spoke IPVPN.";
    }


    identity vpn-topology {
     description
      "Base identity for VPN topology.";
    }
    identity any-to-any {
     base vpn-topology;
     description
      "Identity for any to any VPN topology.";
    }
    identity hub-spoke {
     base vpn-topology;
     description
      "Identity for Hub'n'Spoke VPN topology.";
    }
    identity hub-spoke-disjoint {
     base vpn-topology;
     description
      "Identity for Hub'n'Spoke VPN topology
       where Hubs cannot talk between each other.";
```

```
    }

    identity 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 BiDir 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 BDR 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 LAn are directly connected
        to provider Edge 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
        "icmp v6 protocol type.";
      }
      identity gre {
       base protocol-type;
       description
        "GRE protocol type.";
      }
      identity ipip {
       base protocol-type;
       description
        "IPinIP 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
         admin meaning.";
       }
       choice list-flavor {
        case permit-any {
         leaf permit-any {
          type empty;
          description
          "Allow all sites.";
         }
        }
        case deny-any-except {
         leaf-list permit-site {
          type leafref {
           path "/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
          "Control if
          address translation is required or not.";
        }
        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 in case 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
         "Enable 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 RP
               is provider managed.";
             }
             type boolean;
             default false;
             description
              "If true, redundancy
               mechanism for RP is required.";
            }
            leaf optimal-traffic-delivery {
             when "../enabled = 'true'" {
              description
               "Relevant when RP
                is provider managed.";
             }
             type boolean;
             default false;
             description
              "If true, SP must ensure
               that traffic uses an optimal path.";
            }
            description
             "Parameters for provider managed RP.";
           }

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

   description
    "RendezvousPoint 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 Carrier's Carrier,
         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 country who 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
          is belonging to";
        }
        description
        "List of group-id";
       }
        description
         "Groups the site or site-network-access
         is belonging 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.
          Group values defined here will be inherited
          to all site-network-accesses.";
       }
     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 apply
                  against this particular
                  group-id";
               }
               description
                "List of groups";
              }
             }
             case all-accesses {
```

```
          leaf all-other-accesses {
            type empty;
            description
             "The constraint will apply
             against all other site network
             access
             of this site";
          }
         }
         case all-groups {
          leaf all-other-groups {
            type empty;
            description
             "The constraint will apply
             against all other groups the
             customer
             is managing";
          }
         }
         description
          "Choice for the group definition";
        }
         description
          "The constraint will apply against
          this list of groups";
        }
        description
         "List of constraints";
       }
        description
         "Constraints for placing this site
         network access";
       }

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

   grouping 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
     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
     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
        "Describe flow matching
        criterions.";
      }
      description
       "Flow definition based on criteria.";
     }
     grouping site-service-basic {
      leaf svc-input-bandwidth {
          type uint32;
          units bps;
          description
         "From the PE perspective, the service input
         bandwidth of the connection.";
      }
      leaf svc-output-bandwidth {
          type uint32;
          units bps;
          description
         "From the PE 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 signalling
          protocol between PE and CE.";
        }
        enum "bgp" {
         description
          "Use BGP 3107 as signalling
          protocol between PE and CE.
          In this case, bgp must be also
          configured
          as routing-protocol.
          ";
        }
       }
       description
        "MPLS signalling type.";
      }
      description
       "This container is used when customer provides
       MPLS based services.
       This is used in case of Carrier's
       Carrier.";
     }
     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
      "Need to express marking rules ...";
    }
    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 class must
                be rate
                limited. Expressed as
                percentage of the svc-bw.";
             }
             container latency {
              choice flavor {
               case lowest {
                leaf use-lowest-latency {
                 type empty;
                 description
                  "The traffic class should use
                   the lowest latency path";
                }
               }
               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 lowest jitter path";
               }
              }
              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
               BW in percent of the svc-bw
               available.";
             }
             leaf end-to-end {
              type empty;
              description
               "Used if the bandwidth reservation
               must be done on the MPLS network too";
             }
             description
               "Bandwidth constraint on the traffic
               class";
```

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

        }

       }
       description
       "Qos profile configuration.";
      }
      description
       "QoS configuration.";
     }
     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.
        IPSec may be used as example.";
      }
     }
     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 ...
         Operator specific.";
      }
      leaf strict {
       type boolean;
       default false;
       description
         "define if the requested-type is a preference
         or a strict requirement.";
      }
      description
        "Container for requested type.";
     }
     leaf always-on {
      if-feature always-on;
      type boolean;
      default true;
      description
      "Request for an always on access type.
      This means no Dial access type for
      example.";
     }
     leaf bearer-reference {
      if-feature bearer-reference;
      type string;
      description
        "This is an internal reference for the
        service provider.
```

```
      Used ";
     }
    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 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 through it's ID.";
        }
        leaf metric {
         type uint16;
         description
           "Metric of the sham link.";
        }
        description
          "Creates a shamlink 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
               "Nexthop address to use at 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
               "Nexthop address to use at
               customer side.";
            }
            description "
              List of LAN prefixes for the site.
              ";
           }
           description
            "LAN prefixes from the customer.";
         }
         description
          "Static routing
           specific configuration.";
       }
       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
          "RIP routing specific
           configuration.";
       }

       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
           "VRRP routing specific configuration.";
         }


         description
          "List of routing protocols used
          on the site.
          Need to 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 dhcp allocated";
         }
         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 implementations
          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 server";
       }
       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
         "Provider side address.";
       }
       leaf customer-address {
        type inet:ipv4-address;
        description
         "Customer side address.";
       }
       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 dhcp allocated";
     }
     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 implementations
       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 server";
       }
        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
          "Provider side address.";
        }
        leaf customer-address {
         type inet:ipv6-address;
         description
          "Customer side address.";
        }
        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
               "Service provider well
               known profile.";
            }
            description
              "Service provider well
              known 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
           "Define the OAM 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 has only receivers.";
         }
         enum source-only {
          description
            "The site has only sources.";
         }
         enum source-receiver {
          description
            "The site has both
            sources & 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 like IGMP, 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. Only for
           co-managed case.";
         }
         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 if the site
      is a single VPN site, or multiVPN or ...";
     }
     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-tags to be matched.";
          }
         }
         description
          "Choice for LAN matching type";
        }
        description
         "If used, it permit to split site LANs
         among multiple VPNs.
         If no filter 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 IPVPN.";
        }
        leaf site-role {
         type identityref {
          base site-role;
         }
         default any-to-any-role;
         description
          "Role of the site in the IPVPN.";
        }
```

```
     description
       "List of VPNs the LAN is associated to.";
    }
    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
    "Define maximum-routes for the VRF.";
 }
 description
  "Define 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 attachement.";
     }
     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 attachement.";
     }
     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 cfg.";
      }
      description
       "grouping for extranet VPN configuration.
       Extranet provides a way to interconnect all sites
       from two VPNs in a easy way.";

     }

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

     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 IPVPN.";
      }
     }
     mandatory true;
     description
      "Choice for VPN attachment flavor.";
    }
    description
     "Defines VPN attachment of a site.";
   }
   description
    "Defines the VPN attachment rules
    for a site 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-svc 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 cfg.";
    }
    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";
         }
        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 on how to describe the site 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 cfg.";
 }

 /* 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 modules defined in this document MAY be accessed via the
   RESTCONF protocol [I-D.ietf-netconf-restconf] or NETCONF protocol
   ([RFC6241]. The lowest RESTCONF or NETCONF layer requires that the
   transport-layer protocol provides both data integrity and
   confidentiality, see Section 2 in [I-D.ietf-netconf-restconf] and
   [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 Module (NACM) ([RFC6536]). Other protocols to access this
   YANG module are also required to support the similar mechanism.

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

   o /l3vpn-svc/vpn-services/vpn-service

   o /l3vpn-svc/sites/site

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

11. Contribution

Authors would like to thank Rob Shakir for his major contribution on
the initial modeling and use cases.

12. 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 the contributions to the document.

13. IANA Considerations

IANA is requested to assign a new URI from the IETF XML registry
([RFC3688]). Authors are suggesting the following URI:

    ID: yang:ietf-l3vpn-svc
    URI: urn:ietf:params:xml:ns:yang:ietf-l3vpn-svc
    Filename: [TBD-at-registration]
    Reference: [RFC-to-be]
    Registrant Contact: L3SM WG
    XML: N/A, the requested URI is an XML namespace

This document also requests a new YANG module name in the YANG Module
Names registry ([RFC7950]) with the following suggestion:

    Name: ietf-l3vpn-svc
    Namespace: urn:ietf:params:xml:ns:yang:ietf-l3vpn-svc
    Prefix: l3vpn-svc
    Module:
    Reference: [RFC-to-be]

14. Change Log

14.1. Changes between versions -18 and-19

    o Country code string pattern enforced to ISO ALPHA-2 code.

    o zip-code renamed to postal-code.

    o Added new address-allocation-type: provider-dhcp-slaac.

o Removed transport-constraints and include transport constraints
   (jitter,latency, bandwidth) in the qos-profile.

o qos-profile simplified with more abstraction.

o added target-sites in flow-definition.

14.2. Changes between versions -17 and-18

o Removed TOS from flow matching.

14.3. Changes between versions -16 and-17

o Renamed "vpn-svc" list to "vpn-service".

o Renamed "vpn-policy-list" to "vpn-policies".

o Renamed "management-transport" to "address-family".

o Renamed "multicast-transport" to "address-family".

o Modified cloud access policy using a choice.

o any-to-any-role as default site-role.

o "address-family" is now an enumeration instead of identity.

o cloud-access feature moved to container level.

o Added "address-translation" container for cloud-access.

o Renamed "customer-nat-address" to "customer-address".

o New type ip:address for "customer-address".

o "tree-flavor" moved to leaf-list.

o "bsr-candidate" list moved to "bsr-candidate-address" leaf-list.

o layer becomes mandatory in security-encryption.

o ip-subnet mask range modified.

o multicast transport constraint destination moved to leaf-list.

o lan-prefixes in vpn-policy moved to leaf-list ang tag has been
   renamed "prefixes".

o Added source and destination port range in QoS classification.

o QoS classification uses more existing inet:types.

o Grouping defined for site group list.

14.4. Changes between versions -15 and-16

o Rename "topology" leaf to "vpn-service-topology".

14.5. Changes between versions -13 and-14

o Choice between device reference and location reference.

14.6. Changes between versions -12 and-13

o Removed rip-ng identity (rip container has AF information)

o renamed pe-dhcp to provider-dhcp

o add provider-dhcp-relay identity and container

o BW/MTU is now only under site-network-access

o Add list of location and location ID

o Site-network-access mapped to location Identifier

o Add list of devices (provided by operator) requested by customer

o Some management parameters moved under device list

o Site-network-access mapped to device identifier

14.7. Changes between versions -11 and-12

o Fixing some 'when' statements that prevented compilation.

14.8. Changes between versions -09 and-10

o Removed templates.

o Add site-network-access-type.

o Add a leaf number-of-dynamic-address in case of pe-dhcp
   addressing.

14.9. Changes between versions -08 and-09

   o Add site-vpn-flavor NNI.

14.10. Changes between versions -07 and-08

   o Traffic protection moved to site level.

   o Decouple operational-requirements in two containers.

14.11. Changes between versions -06 and-07

   o Set config false to actual-site-start and stop.

   o Add a container before cloud-access list.

   o Add a container before authorized-sites list.

   o Add a container before denied-sites list.

   o Modified access-diversity modeling.

   o Replacing type placement diversity by an identity.

14.12. Changes between versions -05 and-06

   o Added linecard diverse for site diversity

   o Added a new diversity enum in placement-diversity: none

   o Added state to site location

   o remove reference to core routing model: created new address family
      identities

   o added features

   o Modified bearer parameters

   o Modified union for ipv4/ipv6 addresses to ip-address type

   o Add BSR parameters for multicast

   o Add applications matching for QoS classification

14.13. Changes between versions -04 and-05

   o Modify VPN policy and creating a vpn-policy-list

   o Add VPN policy reference and VPN ID reference under site-network-
      access

14.14. Changes between versions -02 and-03

   o Add extranet-vpn container in vpn-svc

   o Creating top level containers

   o Refine groupings

   o Added site-vpn-flavor

   o qos-profile moved to choice

   o vpn leaf moved to vpn-id in vpn-policy

   o added ordered-by user to qos classification list

   o moved traffic protection to access availability

   o creating a choice in matching filter for VPN policy

   o added dot1p matching field in flow-definition

14.15. Changes between versions -01 and-02

   o A site is now a collection of site-accesses. This was introduced
      to support M to N availability.

   o Site-availability has been removed, replaced by availability
      parameters under site-accesses

   o Added transport-constraints within vpn-svc

   o Add ToS support in match-flow

   o nexthop in cascaded lan as mandatory

   o customer-specific-info deleted and moved to routing protocols

   o customer-lan-connection modified: need prefix and CE address

   o add choice in managing PE-CE addressing

o Simplifying traffic protection

o Refine groupings for vpn-svc

o Removed name in vpn-svc

o id in vpn-svc moved to string

o Rename id in vpn-svc to vpn-id

o Changed key of vpn-svc list to vpn-id

o Add DSCP support in flow definition

o Removed ACL from security

o Add FW for site and cloud access

14.16. Changes between versions -00 and-01

o Creating multiple reusable groupings

o Added mpls leaf in vpn-svc for carrier's carrier case

o Modify identity single to single-site

o Modify site-type to site-role and also child identities.

o Creating OAM container under site and moved BFD in.

o Creating flow-definition grouping to be reused in ACL, QoS ...

o Simplified VPN policy.

o Adding multicast static group to RP mappings.

o Removed native-vpn and site-role from global site cfg, now managed
   within the VPN policy.

o Creating a separate list for site templates.

15. References

15.1. Normative References

   [I-D.ietf-netconf-restconf]
              Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF
              Protocol", draft-ietf-netconf-restconf-18 (work in
              progress), October 2016.

   [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>.

   [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>.

15.2. 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>.

Authors' Addresses

   Stephane Litkowski
   Orange Business Services

   Email: stephane.litkowski@orange.com

   Luis Tomotaki
   Verizon

   Email: luis.tomotaki@verizon.com

   Kenichi Ogaki
   KDDI

   Email: ke-oogaki@kddi.com

Network Working Group   A. Wang
Internet-Draft   China Telecom
Intended status: Standards Track   Y. Cheng
Expires: April 21, 2016   China Unicom
                                                            Y. Zhuang
                                                               Huawei
                                                             D. Zhang
                                                      October 19, 2015

                  Service Automation Management Architecture
              draft-wang-l3sm-service-automation-architecture-01

Abstract

   This document describes a generic architecture for the service
   automation management (SAM) which can be used to deploy service
   across networks models specified in IETF. It describes the basic
   architecture, the components, and interfaces used for service
   automation control and configuration. However, this document does
   not propose protocols or extensions to existing protocols but shows
   how service requests from customer applications can be realized in
   the way more programmable and agile manner and mapped to the
   corresponding protocols and network elements.

Status of This Memo

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

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

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

   This Internet-Draft will expire on April 21, 2016.

Copyright Notice

Table of Contents

1. Introduction

   The purpose of network system configuration is to allow the
   deployment of services requested by customers across networks.
   Essentially, these services are built from a combination of network
   elements and protocol configuration. However, they are presented to
   service customers in a more abstract way.

   To facilitate service automation control and configuration more
   automatic and much simpler manner, service models are proposed and
   specified in IETF to help customers express their service
   requirements as well as operators configure their networks per these
   requirements. As a starting point, L3 VPN Service Model (L3SM)

working group is working on such a service model. [I-D.bogdanovic-netmod-yang-model-classification] discusses YANG model layering which lays a good foundation for L3SM service model work.

The L3VPN Service Model (L3SM) [I-D.ietf-l3sm-l3vpn-service-model] is aimed to provide an abstracted view of a Layer 3 IP VPN service configuration components, which can further break down into the data into specific Network Element models to configure the participating network elements to perform the service . It is more focused on the characteristics of the L3VPN service itself which should be specified in the service data model, while little description of the service model usage, i.e., interacting with other control plane component to configure the network.

This document describes a generic architecture of the service automation management to use service models such as L3SM across the network. The generic SAM architecture is to explain the role of the service model, such as L3 VPN Service Model (L3SM), used in the network and how it can be further processed and deployed onto network nodes to realize the customer service.

This document can help service model developers and network operators better understand the usage of service model and how it can be deployed in their networks.

1.1. Objectives

The purpose of this service-automation management architecture is to provide an outline of how service models from customer applications can be processed and translated into configurations of network elements and protocols, so as to realize the customer requested service.
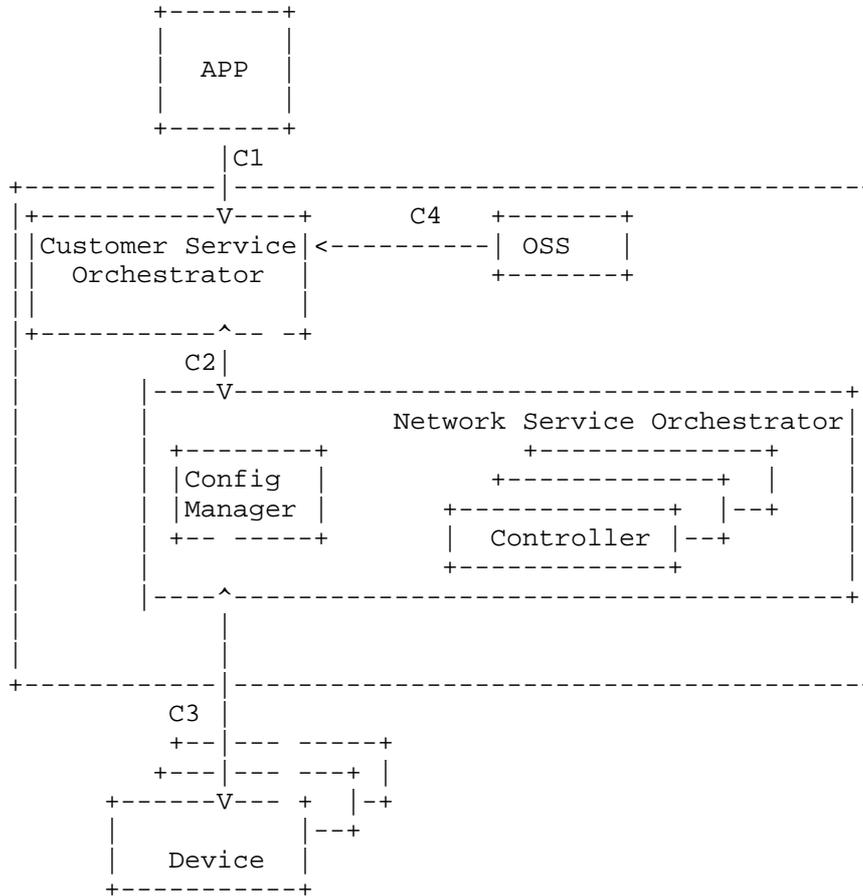
2. Conventions and Terminology

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

3. Generic Architecture for service automation management (SAM)

3.1. SAM architecture

Figure 1 illustrates a generic architecture for the service automation management. The components and functional interfaces are further discussed in Sections 3.2 and 3.3, respectively. It is important to understand that the relationships and interfaces shown

between components in this figure are focused on the instantiation of
service configuration onto network elements.

```
                  +-------+
                  |       |
                  |  APP  |
                  |       |
                  +-------+
                      |C1
 +------------|------------------------------------+
 |+----------V----+    C4   +-------+               |
 ||Customer Service|<----------| OSS   |            |
 ||  Orchestrator |         +-------+               |
 ||               |                                 |
 |+----------^-- -+                                 |
 |          C2|                                     |
 |     |----V------------------------------------+| |
 |     |            Network Service Orchestrator|| |
 |     |  +--------+          +-------------+    || |
 |     |  |Config  |       +------------+   |    || |
 |     |  |Manager |     +------------+  |--+    || |
 |     |  +-- -----+     | Controller |--+       || |
 |     |                 +------------+          || |
 |     |----^------------------------------------+| |
 |     |    |                                      |
 |     |    |                                      |
 +-----|----|------------------------------------+ |
       C3 |                                        
       +--|--- -----+                              
     +---|--- ---+  |                              
 +------V--- +  |-+                                
 |          |--+                                   
 |  Device  |                                      
 +----------+                                      
```

The translation from service model into a set of network
configurations is done by the Customer Service Orchestrator
(CSO),Conf Manager and Network Service Orchestrator. Further, the
translated configuration/control data is signaled using XML based
encoding and YANG module onto involved network elements.

3.2. Functions Components

This section describes the functional components shown as boxes in
Figure 1. The interactions between those components, the functional
interfaces are described in Section 3.3.

3.2.1. APPs

   Within the service automation management architecture, an APP may
   issue high-level service requests to the Customer Service
   Orchestrator which indicates the network service from customer. It
   may also establish policies for the activities of the components
   within the architecture.

   The APP will use service model to carry service characteristics
   parameters and inject thismodel into the Customer Service
   Orchestrator.

3.2.2. Customer Service Orchestrator (CSO)

   The Customer Service Orchestrator is a service component and used by
   service customers to communicate with the network management
   system(e.g.,OSS) to request operations on the network.

   The Customer Service Orchestrator will also interact with the network
   service Orchestrator or Conf Manager to control, operate, and manage
   a network.

   Besides, it might also needs to work with OSS or control plane to get
   network capability information, network requirement information and
   Network Planning information(e.g., route target).

   The Customer Service Orchestrator will use service model sent from
   APP as input and network planning information from OSS or network
   capability information from Control plane and translate them to
   orchestrated configuration (e.g., Technology independent network
   element configuration or Technology dependent Network element
   configuration)of network elements who will be part of the service.

3.2.3. Network Service Orchestrator (NSO)

   The network Service Orchestrator is a configuration component and
   used to control, operate, and manage a network in more scalable way.
   It can interact with the Customer Service Orchestrator to get
   orchestrated configuration of network nodes(e.g., Technology
   independent Network element configuration). In this way, it allows
   the customer service orchestrator to distribute the technology
   independent network configurations for service model in a more
   general way without equipped with various protocols for different
   network device configurations.

   For a Network Service Orchestrator, it can comprise several different
   type of Controller, e.g., ODL Controller, ONOS Controller, ABNO
   Controller, I2RS Controller, SDN Controller), each of which is able

   to install various different protocols described in section 3.4 for
   configuration of different network devices.

   The Conf Manager also can be part of the network service orchestrator
   and used to control, operate, and manage a network as a whole
   dedicatedly Using Netconf protocol.

3.2.4. OSS

   OSS applies to legacy software based platform used by service
   provider to support and manage network operations helping with
   plan,build, operate and optimize the networks. OSS systems support
   network management functions such as network inventory. OSS systems
   will interact with Customer Service Orchestrator and provide Network
   inventory information and network planning informationsuch as route
   target to Customer Service Orchestrator as input to generate
   orchestrated configuration of network elements.

3.2.5. Devices

   Devices can be routers, but also servers (like AAA),while not limited
   to these examples. The configuration of network devices may be done
   by CLI, or by NetConf/RestConf coupled with specific configuration
   YANG data models (BGP, VRF, BFD ...) or any other way by the Network
   Service Orchestrator in section 3.2.3 or the Customer Service
   Orchestrator in section 3.2.2.

3.3. Functional Interfaces

   This section describes the main interfaces between functional
   components involved in service automation control and configuration
   that might be used in an realization of customer services. As noted
   in section 2.2, interfaces shown between components in Figure 1 are
   illustrative and focused on service automation and configuration
   based on service model. It is assumed that existing protocols can
   provide all of the capabilities. The discussion of new protocols for
   these capabilities is out of scope.

3.3.1. C1: Service Level interface

   The north-bound Service Level interfaces to the customer service
   orchestrator (CSO) are used by applications. With the service level
   interface, customers can request customer network service with
   various service requirements needed in the network. The interface
   will also need to be able to report the asynchronous completion of
   service requests and convey changes in the status of services, but
   these status information are not necessary to be carried by service
   model.

3.3.2. C2: Network Wide Interface

   The network Level orchestrator is used to provide network
   configuration on the network for customer services by operators. In
   this case, the network Wide interface from the customer service
   orchestrator is introduced.

   After the customer service orchestrator maps the service model of the
   requested network service into network configuration data, it sends
   out the configuration data onto network service orchestrator via the
   network wide interface while leaving the detailed protocol-related
   configuration of involved network devices, such as routers and AAA,
   to the network service orchestrator itself. Note that Network Wide
   Interface can be an internal interface between Customer Service
   Orchestrator and Network service orchestrator. That is to say,
   Customer Service orchestrator is collocated with Network service
   orchestrator or Conf Manager.

3.3.3. C3: Device Level Interface

   Upon receiving a orchestrated network configuration data from the
   Network Service Orchstrator, the network service orchestrator or Conf
   Manager further configures the managed network devices by using
   corresponding protocols (See section 3.4) via the device level
   interfaces.

3.3.4. C4: Network Level OSS facing interface

   By using these configuration interfaces, the Customer Service
   Orchestrator can interact with OSS and get planning information or
   verify whether the network resource can meet service requirements in
   the service model by looking up network capability provided by OSS
   and determine appropriate network level configuration parameters.

3.4. Candidate Network Configuration Management Protocols

   As noted in 3.2, the Customer Service Orchestrator,Network Service
   Orchestrator, Conf Manager is to install corresponding protocols to
   exchange information with involved network devices. Many protocols
   already exist to perform these functions, including the following:

   o SNMP [RFC3412]

   o The Network Configuration Protocol (NETCONF) [RFC6241]

   o RESTCONF [RESTCONF]

   o The General Switch Management Protocol (GSMP) [RFC3292]

   o ForCES [RFC5810]

   o OpenFlow [ONF]

   o PCEP [PCE-Init-LSP]

3.5. Relation between service model and Network element model

   Service Models are created to describe the characteristics of a
   service, as agreed upon with consumers of that service while Network
   Element Model describes the configuration parameters of a network
   protocol and features for a participanting network device. In
   addition the network element model describe operation state of the
   network protocol for corresponding network device. The Network
   element Models can be further broken down into Technology independent
   YANG model and Technology specific YANG model. Unlike Network
   element models, the service models can not be directly implemented
   into network element. For more details on difference between service
   model and network model, please refer to [I-D.bogdanovic-netmod-yang-
   model-classification].

4. Security Considerations

   TBD.

5. IANA Considerations

   TBD.

6. Normative References

   [I-D.bogdanovic-netmod-yang-model-classification]
              Bogdanovic, D., Claise, B., and C. Moberg, "YANG Model
              Classification", draft-bogdanovic-netmod-yang-model-
              classification-04 (work in progress), October 2015.

   [I-D.ietf-l3sm-l3vpn-service-model]
              Litkowski, S., Shakir, R., Tomotaki, L., and K. D'Souza,
              "YANG Data Model for L3VPN service delivery", draft-ietf-
              l3sm-l3vpn-service-model-01 (work in progress), August
              2015.

   [I-D.ietf-netconf-restconf]
              Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF
              Protocol", draft-ietf-netconf-restconf-08 (work in
              progress), October 2015.

   [I-D.ietf-pce-pce-initiated-lsp]
             Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP
             Extensions for PCE-initiated LSP Setup in a Stateful PCE
             Model", draft-ietf-pce-pce-initiated-lsp-04 (work in
             progress), April 2015.

   [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>.

   [RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the
             Network Configuration Protocol (NETCONF)", RFC 6020,
             October 2010.

   [RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A.
             Bierman, "Network Configuration Protocol (NETCONF)",
             RFC 6241', June 2011.

Authors' Addresses

   Aijun Wang
   China Telecom
   No.118,Xizhimenneidajie,Xicheng District
   Beijing 100035
   China

   Email: wangaj@ctbri.com.cn

   Ying Cheng
   China Unicom
   P.R. China

   Email: chengying10@chinaunicom.cn

   Yan Zhuang
   Huawei
   101 Software Avenue, Yuhua District
   Nanjing, Jiangsu 210012
   China

   Email: zhuangyan.zhuang@huawei.com

Dacheng Zhang

Email: dacheng.zhang@gmail.com