draft-ietf-l3sm-l3vpn-service-model

S. Litkowski
R. Shakir
L. Tomotaki
K. D’Souza

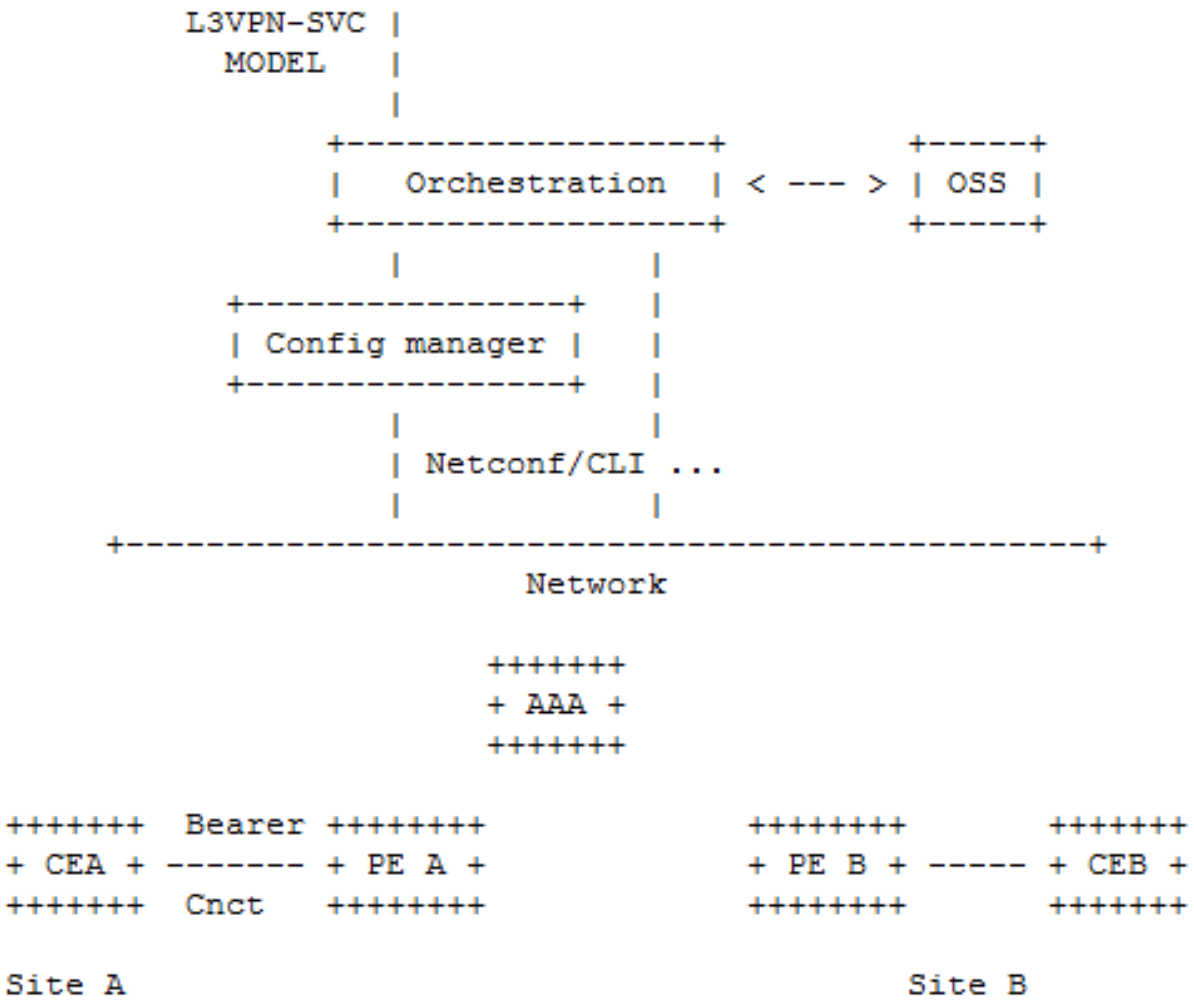
Context

- A lot of YANG models for network elements and protocols are in progress
- Need also service models
- Let’s start with Layer 3 VPN « famous » service
- L3SM WG set up to follow the work (short live WG)

A service model, not a configuration model

- Service model provides an abstraction of customer requirements to build the service
- No bits and bytes regarding protocol and element detailed configuration
- Focus : what the customer wants in abstracted terms

Service model and config models working together

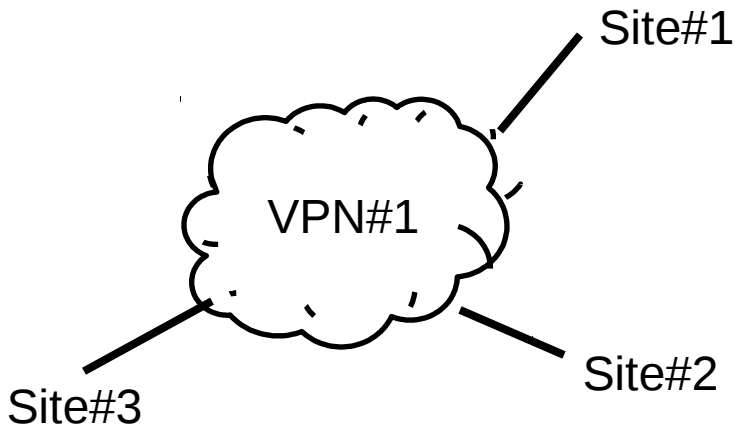


Note …

- We started from PE-Based L3VPN service …
- The model may need to be extended/modified to support all types of L3VPN.

Design of the model

- Two main blocks :
  - vpn-svc :
    - Describe a VPN and its associated services (Cloud, multicast …)
  - sites :
    - Describe a VPN site



Design of the model : vpn-svc

- VPNs are identified by a unique name
- We also provide an optional ID
- We cannot use the customer-name as a key, as a customer may have multiple VPNs

- A VPN has a specific topology :
  - Anytoany, Hub&Spoke, Hub&Spoke disjoint

```
module: ietf-l3vpn-svc
   +--rw l3vpn-svc
      +--rw vpn-svc* [name]
      |  +--rw name                string
      |  +--rw id?                 uint32
      |  +--rw customer-name?      string
      |  +--rw topology?           identityref
      |  +--rw cloud-access* [cloud-identifier]
      |  |  +--rw cloud-identifier          string
      |  |  +--rw authorized-sites* [site-id]
      |  |  |  +--rw site-id       leafref
      |  |  +--rw denied-sites* [site-id]
      |  |  |  +--rw site-id       leafref
      |  |  +--rw nat-enabled?              boolean
      |  |  +--rw customer-nat-address?     inet:ipv4-address
      |  +--rw multicast
      |     +--rw tree-flavor*             identityref
      |     +--rw rp
      |     |  +--rw ipv4-address?     inet:ipv4-address
      |     |  +--rw ipv6-address?     inet:ipv6-address
      |     +--rw rp-discovery?            identityref
      |     +--rw anycast-rp-location*     string
```

Design of the model : vpn-svc

- Services can be added on the VPN :
  - Cloud access :
    - Access to any Cloud service Provider
    - CSP identified by an internal ID (local administrative identificator)
    - Some sites can be registered to access to the CSP
    - NAT to CSP is possible
  - Multicast :
    - Allow to enable multicast traffic on the VPN
    - Some strong coordination with customer parameters required (type of tree …)

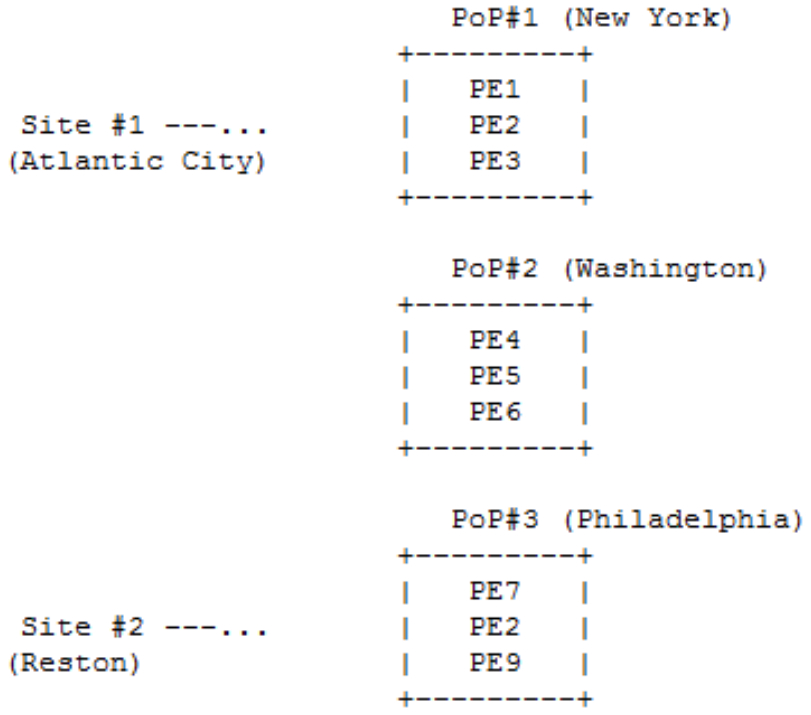
Design of the model : sites

- Site parameters :
  - ID
  - Type of site (in the VPN topology) -> Hub, Spoke …
  - Possibility to schedule site enabling
  - Location of the site (address)
  - Diversity parameters :
    - Do I need to have sites not connected on the same provider edge ?
  - Security parameters (encryption …)
  - Availability parameters :
    - do I need a backup ? Or loadsharing ?
  - Type of attachment (bearer, IP layer, routing …)
  - Services :
    - QoS, Bandwidth, MTU, protection …
  - Management :
    - Is it a comanaged site ?
  - VPN policy
  - Customer specific information

```
+--rw sites* [site-id]
   +--rw template?                        boolean
   +--rw site-id                          string
   +--rw native-vpn?                      leafref
   +--rw site-type?                       identityref
   +--rw apply-template?                  leafref
   +--rw requested-site-start?            yang:date-and-time
   +--rw requested-site-stop?             yang:date-and-time
   +--rw actual-site-start?               yang:date-and-time
   +--rw actual-site-stop?                yang:date-and-time
   +--rw location
   |     ...
   +--rw site-diversity
   |     ...
   +--rw security
   |     ...
   +--rw availability
   |     ...
   +--rw attachment
   |     ...
   +--rw service
   |     ...
   +--rw management
   |     ...
   +--rw vpn-policy
   |     ...
   +--rw maximum-routes
   |     ...
   +--rw customer-specific-information
         ...
```

Site location

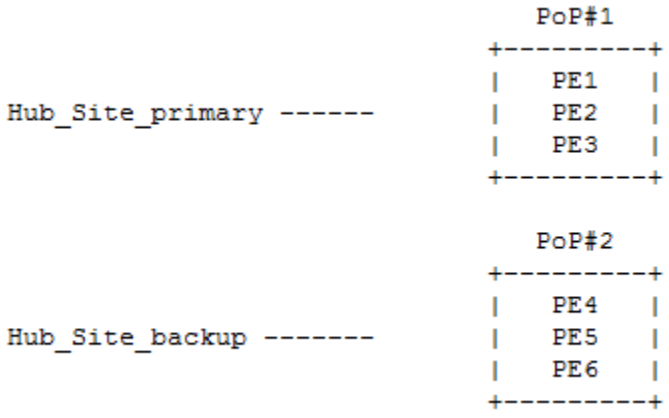
- The address of the site would permit to the SP OSS to find the appropriate provider edge to place the customer access.

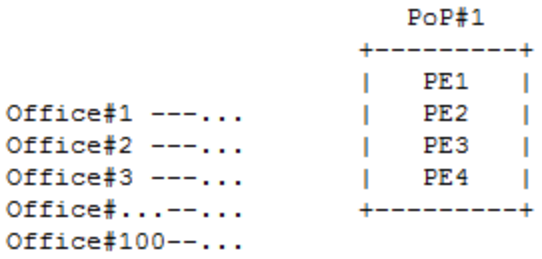


Site diversity

- When placing accesses onto network elements, customer may want to avoid some sites to share fate.

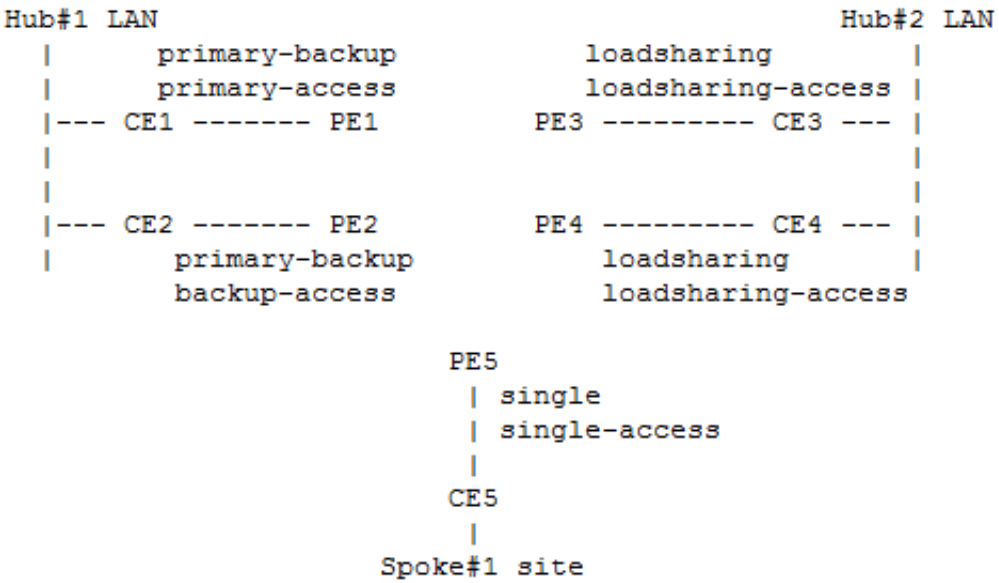
- Two proposed options : PoP diversity, PE diversity





Site availability

- Increase availability of the site access
- Three options :
  - Single : no redundancy (basic)
  - Primary-backup : dual homing scenario, with traffic primarly going to one site
  - Loadsharing : multihoming scenario
- Each access has a function in the availability scenario through the « access-type » : single-access, primary-access,backup-access,loadsharing-access



Site attachment

- Attachment = customer connection to the SP
- Required parameters from the customer or external systems :
  - Some physical parameters (bearer)
  - IP address allocation
  - Type of routing
  - Fast failure detection or not

```
+--rw attachment
|  +--rw apply-template?   leafref
|  +--rw bearer
|  |  +--rw type?               string
|  |  +--rw bearer-reference?   string
|  +--rw connection
|     +--rw ipv4
|     |  +--rw address-allocation-type?   identityref
|     |  +--rw subnet-prefix?             inet:ipv4-prefix
|     +--rw ipv6
|     |  +--rw address-allocation-type?   string
|     |  +--rw subnet-prefix?             inet:ipv6-prefix
|     +--rw routing-protocols* [type]
|        +--rw type      identityref
|        +--rw ospf
|        |  +--rw address-family*   identityref
|        |  +--rw area-address?     yang:dotted-quad
|        |  +--rw metric?           uint16
|        |  +--rw sham-link* [target-site]
|        |     +--rw target-site    leafref
|        |     +--rw metric?        uint16
|        +--rw bgp
|        |  +--rw address-family*   identityref
|        +--rw static
|        |  +--rw address-family*   identityref
|        +--rw rip
|        |  +--rw address-family*   identityref
|        +--rw vrrp
|        |  +--rw address-family*   identityref
|        +--rw bfd
|           +--rw bfd-enabled?    boolean
|           +--rw (holdtime)?
|              +--:(profile)
|              |     ...
|              +--:(fixed)
|                    ...
```

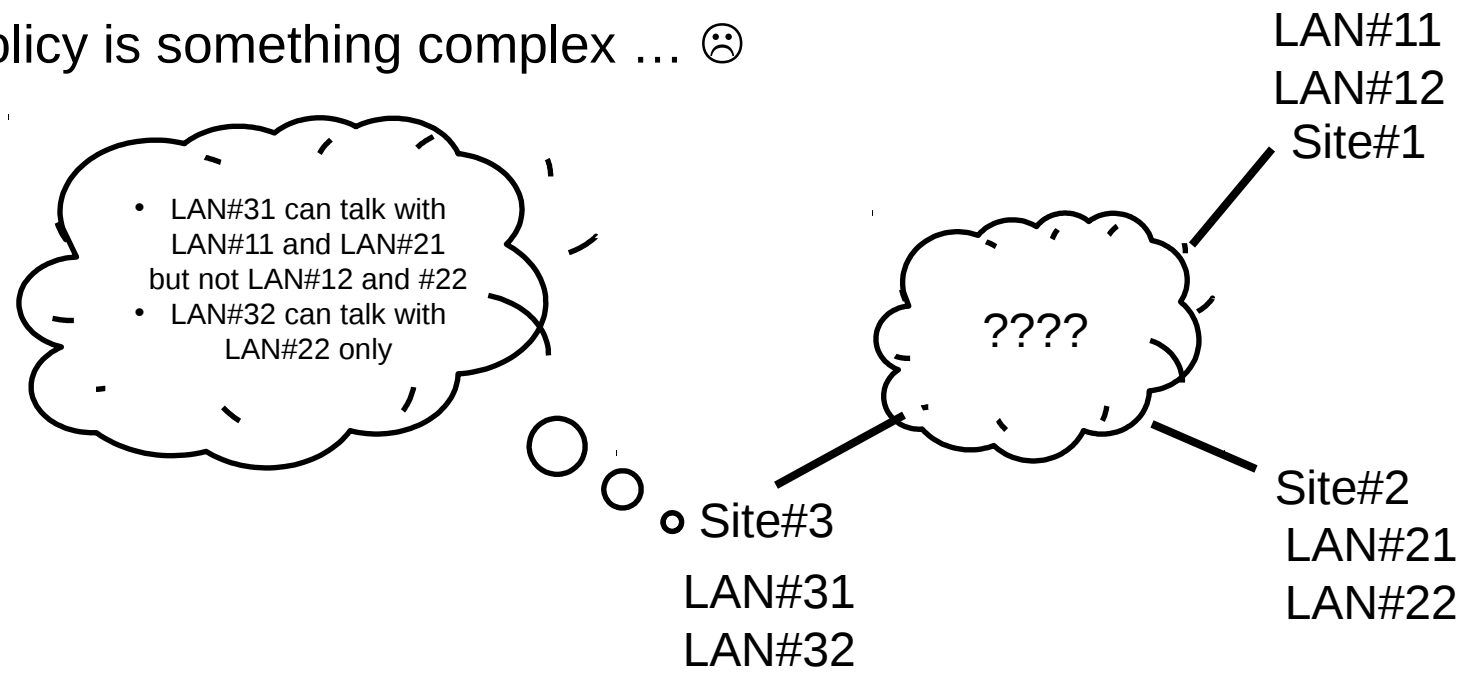
Site services

- Defining QoS requirements : customized or SP profile
- Defining BW (may be asymetric)
- Defining if protection is required
- Defining if MPLS or multicast forwarding is required

```
+--rw service
|  +--rw apply-template?          leafref
|  +--rw qos
|  |  +--rw qos-classification-policy
|  |  |  +--rw rules* [id]
|  |  |     +--rw id                    uint16
|  |  |     +--rw match
|  |  |     |  +--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?       uint16
|  |  |     |  +--rw l4-dst-port?       uint16
|  |  |     |  +--rw l4-protocol?       union
|  |  |     +--rw target-class-id?   string
|  |  +--rw std-qos-profile?             string
|  |  +--rw custom-qos-profile
|  |     +--rw class* [class-id]
|  |        +--rw class-id                 string
|  |        +--rw rate-limit?              uint8
|  |        +--rw priority-level?          uint8
|  |        +--rw guaranteed-bw-percent?   uint8
|  +--rw svc-input-bandwidth?    uint32
|  +--rw svc-output-bandwidth?   uint32
|  +--rw svc-mtu?                uint16
|  +--rw traffic-protection
|  |  +--rw link-local-protection?    boolean
|  |  +--rw node-local-protection?    boolean
|  |  +--rw node-global-protection?   boolean
|  +--rw mpls
|  |  +--rw signalling-type?   enumeration
|  +--rw multicast
|     +--rw site-type?   enumeration
```

VPN policy

- A site can be part of multiple VPNs
- Moreover some LANs of a site can be part of some VPNs, while some other LAN can be part of others.
- VPN policy is something complex … ☹



VPN policy

- We introduce the notion of native VPN
- A site belongs to ONLY one native VPN : this does not mean that the site belongs to only one VPN !
- Base behavior :
  - All prefixes of the site will be able to reach other prefixes of other sites in the native VPN according to the VPN topology (any to any, hub & spoke …)
- More complex scenarios are created by using vpn-policy

VPN policy

- Why not multiple native VPNs ?
  - This is causing issues if two « native » VPNs have different topologies and the site has a different role in those topologies.
  - Example : site #1 belongs to VPN A (H&S) and is a Hub, and belongs to VPN B (H&S) and is a spoke ☹
- Native VPN does not prevent a site to belongs to multiple VPN … see next slide …

VPN policy

- VPN policy defines a set of communication rules
- No need of VPN policy if only communication rules of the VPN native are used. VPN policy is there to create more complex rules
- Today we use import/export concept but maybe not enough abstracted or do we need to rely on policy model in RTGW(

```
+--rw vpn-policy
|  +--rw import-policy
|  |  +--rw vpn*   leafref
|  +--rw export-policy
|     +--rw entries* [id]
|        +--rw id              uint32
|        +--rw lan-prefixes
|        |  +--rw ipv4-lan-prefixes* [lan]
|        |  |  +--rw lan    inet:ipv4-prefix
|        |  +--rw ipv6-lan-prefixes* [lan]
|        |     +--rw lan    inet:ipv6-prefix
|        +--rw lan-tag*        string
|        +--rw vpn*            leafref
```

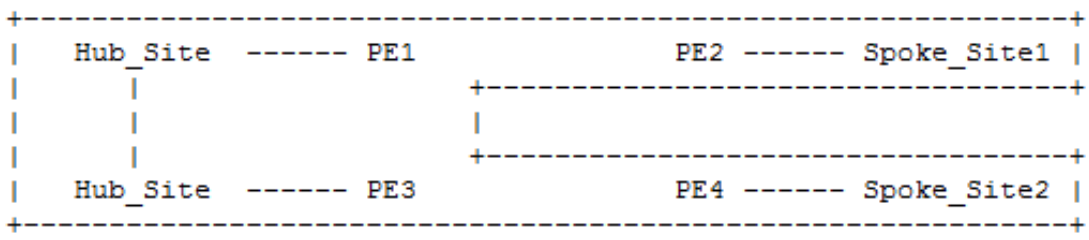
Customer specific information

- To ensure proper configuration through the config models, some parameters from the customers may be required

```
+--rw customer-specific-information
   +--rw name?                        string
   +--rw autonomous-system?           uint32
   +--rw interface?                   string
   +--rw customer-lan-connection* [address]
   |  +--rw address          union
   |  +--rw lan-protocol?    identityref
   +--rw cascaded-lan-prefixes
      +--rw ipv4-lan-prefixes* [lan]
      |  +--rw lan          inet:ipv4-prefix
      |  +--rw lan-tag?     string
      |  +--rw next-hop?    inet:ipv4-address
      +--rw ipv6-lan-prefixes* [lan]
         +--rw lan          inet:ipv6-prefix
         +--rw lan-tag?     string
         +--rw next-hop?    inet:ipv6-address
```

Mapping service model to config model

- Example :
  - We want to create Spoke_site#1 in this VPN :




```
<vpn-svc>
    <name>VPN1</name>
    <id>12456487</id>
    <customer-name>CUSTOMER1</customer-name>
    <topology>hub-spoke</topology>
</vpn-svc>
```

Mapping service model to config model

```
<sites>
    <site-id>Spoke_Site1</site-id>
    <native-vpn>VPN1</native-vpn>
    <site-type>spoke-site</site-type>
    <location>
        <city-code>NY</city-code>
        <country-code>US</country-code>
    </location>
    <attachment>
        <connection>
            <ipv4>
                <subnet-prefix>203.0.113.0/30</subnet-prefix>
            </ipv4>
            <routing-protocol>
                <type>bgp</type>
                <bgp>
                    <address-family>ipv4-unicast</address-family>
                </bgp>
            </routing-protocol>
        </connection>
    </attachment>
    <management>
        <type>provider-managed</type>
        <management-transport>ipv4-unicast</management-transport>
        <address>10.46.1.1</address>
    </management>
    <service>
        <svc-input-bandwidth>450000000</svc-input-bandwidth>
        <svc-output-bandwidth>450000000</svc-output-bandwidth>
    </service>
    <customer-specific-information>
        <customer-lan-connection>
            <address>192.0.2.254</address>
            <lan-protocol>ipv4-unicast</lan-protocol>
        </customer-lan-connection>
        <cascaded-lan-prefixes>
            <ipv4-lan-prefixes>
                <lan>198.51.100.0/30</lan>
                <nexthop>192.0.2.253</nexthop>
            </ipv4-lan-prefixes>
            <ipv4-lan-prefixes>
                <lan>198.51.100.4/30</lan>
                <nexthop>192.0.2.253</nexthop>
            </ipv4-lan-prefixes>
        </cascaded-lan-prefixes>
    </customer-specific-information>
</sites>
```

```
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
!
interface Ethernet1/1/0.10
 encapsulation dot1q 10
 ip vrf forwarding Customer1
 ip address 203.0.113.1 255.255.255.252 <---- Comes from
                                              subnet-prefix
 ip access-group STD-PROTECT-IN     <---- Standard SP config
!
router bgp 100
 address-family ipv4 vrf Customer1
  neighbor 203.0.113.2 remote-as 65000   <---- Comes from
                                               subnet-prefix
                                               and allocated CE ASN
  neighbor 203.0.113.2 route-map STD in  <---- Standard SP config
  neighbor 203.0.113.2 filter-list 10 in <---- Standard SP config
!
ip route vrf Customer1 203.0.113.254 255.255.255.255 203.0.113.2
! Static route for provider administration of CE
!

Example of generated CE configuration :

interface Loopback10
 description "Administration"
 ip address 203.0.113.254 255.255.255.255
!
interface FastEthernet10
 description "WAN"
 ip address 203.0.113.2 255.255.255.252 <---- Comes from
                                              subnet-prefix
!
interface FastEthernet11
 description "LAN"
 ip address 192.0.2.254 255.255.255.252 <---- Comes from
                                        customer-lan-connection
!
router bgp 65000
  redistribute static route-map STATIC2BGP <---- Standard SP
                                                 configuration
  neighbor 203.0.113.1 remote-as 100     <---- Comes from
                                               subnet-prefix
                                               and allocated CE ASN
!
route-map STATIC2BGP permit 10
 match tag 10
!
ip route 198.51.100.0 255.255.255.252 192.0.2.253 tag 10
ip route 198.51.100.4 255.255.255.252 192.0.2.253 tag 10
```

Site templates

- VPNs may have many sites, and some sites may share the same description
- We can use templates to refer to some shared configuration

Site templates

- Template definition :
  - Create a site with « template=true »
    - No need to detail all the parameters, just describe the ones you want to inherit
  - Apply template : Template can be applied at :
    - top level of the site (inherit all the config from the template)
    - Security section (only security section is inherited)
    - Attachment section
    - Service section
    - A parameter defined in a real site must override inherited parameter

Not finished … next steps …

- We still need to work on :
  - Comments from the list :
    - Do we need externalize Cloud accesses and multicast from VPN ?
    - Some wordings to be changed …
  - Security parameters
  - VPN policy ?
  - Need to review if the current proposal fits any L3VPN rather than PE-Based only
- Operational states ?
- What about interAS consideration ?
  - In my mind, nothing to do … but need to be discussed !
- What about Hybrid VPNs (public+private sites) ?
- What about value added services ? (DDoS, antivirus, DPI, …)
- Anything else ?