Authors: Y. Lee, Ed.   D. Dhody, Ed.
         Samsung Electronics   Huawei Technologies
         D. Ceccarelli   I. Bryskin   B. Yoon
         Ericsson   Individual   ETRI

A YANG Data Model for VN Operation

Abstract

   This document provides a YANG data model generally applicable to any
   mode of Virtual Network (VN) operation.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document provides a YANG [RFC7950] data model generally
applicable to any mode of Virtual Network (VN) operation.

The VN model defined in this document is applicable in generic sense
as an independent model in and of itself. The VN model defined in
this document can also work together with other customer service
models such as L3SM [RFC8299], L2SM [RFC8466] and L1CSM [I-D.ietf-
ccamp-l1csm-yang] to provide a complete life-cycle service
management and operations.

The YANG model discussed in this document basically provides the following:

  *Characteristics of Access Points (APs) that describe customer's
   end point characteristics;

  *Characteristics of Virtual Network Access Points (VNAP) that
   describe how an AP is partitioned for multiple VNs sharing the AP
   and its reference to a Link Termination Point (LTP) of the
   Provider Edge (PE) Node;

  *Characteristics of Virtual Networks (VNs) that describe the
   customer's VN in terms of multiple VN Members comprising a VN,
   multi- source and/or multi-destination characteristics of the VN
   Member, the VN's reference to TE-topology's Abstract Node;

The actual VN instantiation and computation is performed with
Connectivity Matrices sub-module of TE-Topology Model [RFC8795]
which provides TE network topology abstraction and management
operation. Once TE-topology Model is used in triggering VN
instantiation over the networks, TE-tunnel [I-D.ietf-teas-yang-te]
Model will inevitably interact with TE-Topology model for setting up
actual tunnels and LSPs under the tunnels.

Abstraction and Control of Traffic Engineered Networks (ACTN)
describes a set of management and control functions used to operate
one or more TE networks to construct virtual networks that can be
represented to customers and that are built from abstractions of the
underlying TE networks [RFC8453]. ACTN is the primary example of the
usage of the VN YANG model.

Sections 2 and 3 provide the discussion of how the VN YANG model is
applicable to the ACTN context where Virtual Network Service (VNS)
operation is implemented for the Customer Network Controller (CNC)-
Multi-Domain Service Coordinator (MSDC) interface (CMI).

The YANG model on the CMI is also known as customer service model in
[RFC8309]. The YANG model discussed in this document is used to
operate customer-driven VNs during the VN instantiation, VN
computation, and its life-cycle service management and operations.

The VN operational state is included in the same tree as the
configuration consistent with Network Management Datastore
Architecture (NMDA) [RFC8342]. The origin of the data is indicated
as per the origin metadata annotation.

1.1. Terminology

Refer to [RFC8453], [RFC7926], and [RFC8309] for the key terms used
in this document.

1.1.1. Requirements Language

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

1.2. Tree diagram

A simplified graphical representation of the data model is used in
Section 5 of this this document. The meaning of the symbols in these
diagrams is defined in [RFC8340].

1.3. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects
are prefixed using the standard prefix associated with the
corresponding YANG imported modules, as shown in Table 1.

Prefix	YANG module	Reference
vn	ietf-vn	[RFCXXXX]
yang	ietf-yang-types	[RFC6991]
nw	ietf-network	[RFC8345]
nt	ietf-network-topology	[RFC8345]
te-types	ietf-te-types	[RFC8776]
tet	ietf-te-topology	[RFC8795]

Table 1: Prefixes and corresponding YANG
modules

Note: The RFC Editor will replace XXXX with the number assigned to
the RFC once this draft becomes an RFC.

2. Use-case of VN YANG Model in the ACTN context

In this section, ACTN is being used to illustrate the general usage
of the VN YANG model. The model presented in this section has the
following ACTN context.

```
                +-------+
                | CNC   |
                +-------+
                    |
                    |     VN YANG + TE-topology YANG
                    |
          +------------------------+
          |          MDSC          |
          +------------------------+
```
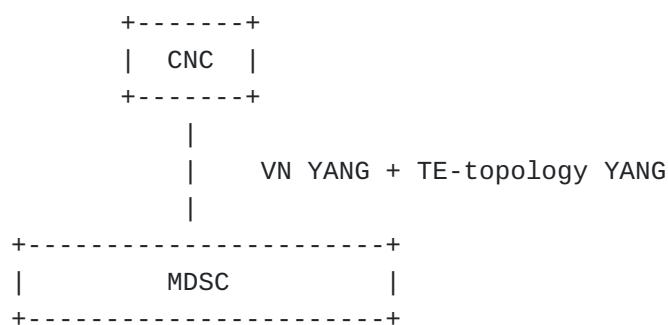
                     Figure 1: ACTN CMI

   Both ACTN VN YANG and TE-topology models are used over the CMI to
   establish a VN over TE networks.

2.1. Type 1 VN

   As defined in [RFC8453], a Virtual Network is a customer view of the
   TE network. To recapitulate VN types from [RFC8453], Type 1 VN is
   defined as follows:

   The VN can be seen as a set of edge-to-edge abstract links (a Type 1
   VN). Each abstract link is referred to as a VN member and is formed
   as an end-to-end tunnel across the underlying networks. Such tunnels
   may be constructed by recursive slicing or abstraction of paths in
   the underlying networks and can encompass edge points of the
   customer's network, access links, intra-domain paths, and inter-
   domain links.

   If we were to create a VN where we have four VN-members as follows:

               VN-Member 1   L1-L4
               VN-Member 2   L1-L7
               VN-Member 3   L2-L4
               VN-Member 4   L3-L8

      Where L1, L2, L3, L4, L7 and L8 correspond to a Customer End-
      Point, respectively.

   This VN can be modeled as one abstract node representation as
   follows in Figure 2:

```
              +---------------+
      L1 ------|               |------ L4
      L2 ------|     AN 1      |------ L7
      L3 ------|               |------ L8
              +---------------+
```

         Figure 2: Abstract Node (One node topology)

Modeling a VN as one abstract node is the easiest way for customers
to express their end-to-end connectivity; however, customers are not
limited to express their VN only with one abstract node.

2.2. Type 2 VN

For some VN members of a VN, the customers are allowed to configure
the actual path (i.e., detailed virtual nodes and virtual links)
over the VN/abstract topology agreed mutually between CNC and MDSC
prior to or a topology created by the MDSC as part of VN
instantiation. Type 1 VN is a higher abstraction of a Type 2 VN.

If a Type 2 VN is desired for some or all of VN members of a type 1
VN (see the example in Section 2.1), the TE-topology model can
provide the following abstract topology (that consists of virtual
nodes and virtual links) which is built under the Type 1 VN.

```
          +------------------------------------------------+
          |                 S1                    S2       |
          |                 0---------------0              |
          |        _____/ _____         \            |
          |     /                   \           \         |
          |S3 /                      \ S4      \ S5       |
    L1----|-0-----------------------0---------0-----------|------L4
          |    \                     \          \         |
          |     \                     \          \        |
          |      \ S6                  \ S7       \ S8     |
          |       0      ---------------0---------0-------|------L7
          |      / \    /               \   ___/         |
          |S9  /   \ /S10                \ /             |
    L2-----|---0-----0--------------------0-------------|------L8
          |  /                          S11             |
    L3-----|--                                           |
          |                                              |
          +------------------------------------------------+
```
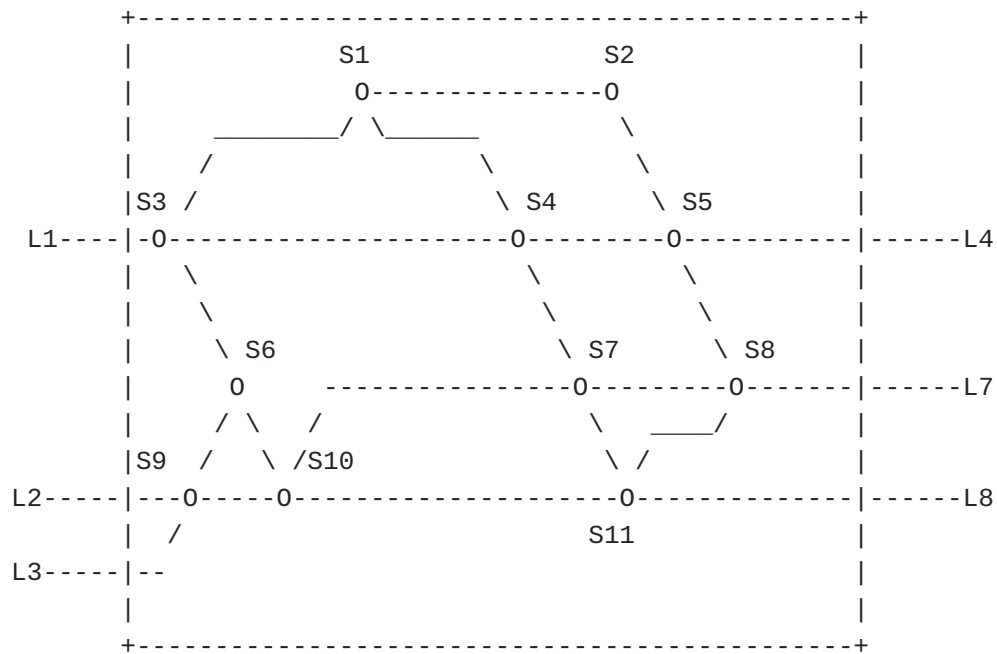
Figure 3: Type 2 topology

As you see from Figure 3, the Type 1 abstract node is depicted as a
Type 1 abstract topology comprising of detailed virtual nodes and
virtual links.

As an example, if VN-member 1 (L1-L4) is chosen to configure its own
path over Type 2 topology, it can select, say, a path that consists
of the ERO {S3,S4,S5} based on the topology and its service
requirement. This capability is enacted via TE-topology
configuration by the customer.

3. High-Level Control Flows with Examples

3.1. Type 1 VN Illustration

   If we were to create a VN where we have four VN-members as follows:

            VN-Member 1   L1-L4
            VN-Member 2   L1-L7
            VN-Member 3   L2-L4
            VN-Member 4   L3-L8

   Where L1, L2, L3, L4, L7 and L8 correspond to Access Points.

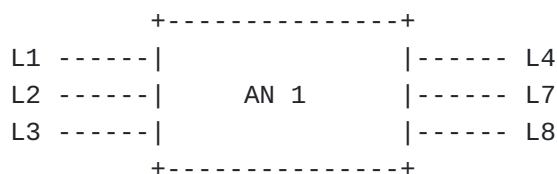   This VN can be modeled as one abstract node representation as
   follows:

```
                 +---------------+
        L1 ------|               |------ L4
        L2 ------|     AN 1      |------ L7
        L3 ------|               |------ L8
                 +---------------+
```

   If this VN is Type 1, the following diagram shows the message flow
   between CNC and MDSC to instantiate this VN using VN and TE-Topology
   Models.

```
         +--------+                      +--------+
         |  CNC   |                      |  MDSC  |
         +--------+                      +--------+
              |                              |
              |                              |
CNC POST TE-topo |  POST /nw:networks/nw:network/   |
model(with Conn. |  nw:node/te-node-id/             |
Matrix on one    |  tet:connectivity-matrices/      |
Abstract node    |  tet:connectivity-matrix         |
              |------------------------------->|
              |                    HTTP 200    |
              |<-------------------------------|
              |                              |
CNC POST the     |  POST /virtual-network         |
VN identifying   |------------------------------->| If there is
AP, VNAP and VN- |                              | multi-src/dest
Members and maps |                              | then MDSC
to the TE-topo   |                    HTTP 200  | selects a
              |<-------------------------------| src or dest
              |                              | and update
              |                              | VN YANG
CNC GET the      |  GET /virtual-network          |
VN YANG status   |------------------------------->|
              |                              |
              |  HTTP 200 (VN with status:     |
              |  selected VN-members           |
              |  in case of multi s-d)         |
              |<-------------------------------|
              |                              |
```

3.2. Type 2 VN Illustration

   For some VN members, the customer may want to "configure" explicit
   routes over the path that connects its two end-points. Let us
   consider the following example.

      VN-Member 1 L1-L4 (via S3, S4, and S5)

      VN-Member 2 L1-L7 (via S3, S4, S7 and S8)

      VN-Member 3 L2-L7 (via S9, S10, and S11)

      VN-Member 4 L3-L8 (via S9, S10 and S11)

   Where the following topology is the underlay for Abstraction Node 1
   (AN1).

```
                          AN1
          .............................................
          .                S1                 S2          .
          .              O--------------O                 .
          .        _____/ _____          \           .
          .       /              \             \          .
          .  S3/                   \ S4       \ S5        .
    L1-----.-O----------------------O---------O-------.---------L4
          .    \                    \          \         .
          .     \                    \          \        .
          .      \ S6                  \ S7       \ S8 .
          .        O        ---------------O---------O---.---------L5
          .       / \   /                   \    ____/ \__._____L6
          .S9  /    \ /S10                    \ /        .
    L2------.---O-----O--------------------O----------.---------L7
          .    /                          S11_____._____L8
     L3-----.--                                        .
          ..........................................
```
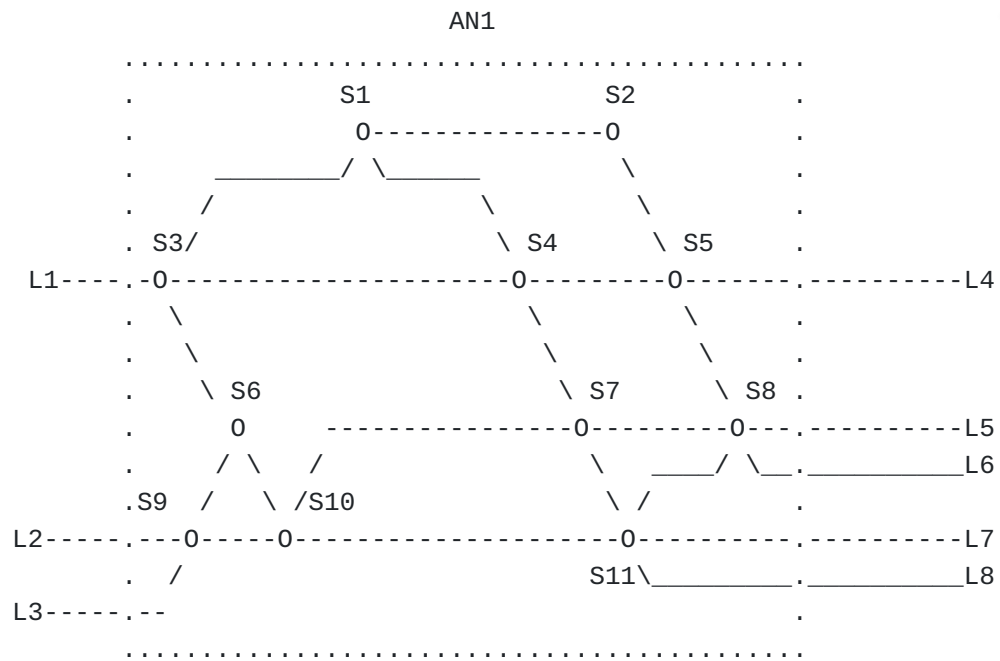
There are two options depending on whether CNC or MDSC creates the
single abstract node topology.

Case 1:

If CNC creates the single abstract node topology, the following
diagram shows the message flow between CNC and MDSC to instantiate
this VN using VN and TE-Topology Model.

```
           +--------+                       +--------+
           |  CNC   |                       |  MDSC  |
           +--------+                       +--------+
                |                                |
                |                                |
CNC POST TE-topo |  POST /nw:networks/nw:network/ |
model(with Conn. |  nw:node/te-node-id/tet:connectivity-   |
Matrix on one    |  matrices/tet:connectivity-matrix       |
Abstract node and|--------------------------------------->|
Explicit paths in|                                |
the conn. matrix)|                HTTP 200        |
                |<---------------------------------------|
                |                                |
CNC POST the     |  POST /virtual-network         |
VN identifying   |--------------------------------------->|
AP, VNAP and VN- |                                |
Members and maps |                                |
to the TE-topo   |                    HTTP 200    |
                |<---------------------------------------|
                |                                |
                |                                |
CNC GET the      |  GET /virtual-network          |
VN YANG status   |--------------------------------------->|
                |                                |
                |  HTTP 200 (VN with status)     |
                |<---------------------------------------|
                |                                |
```
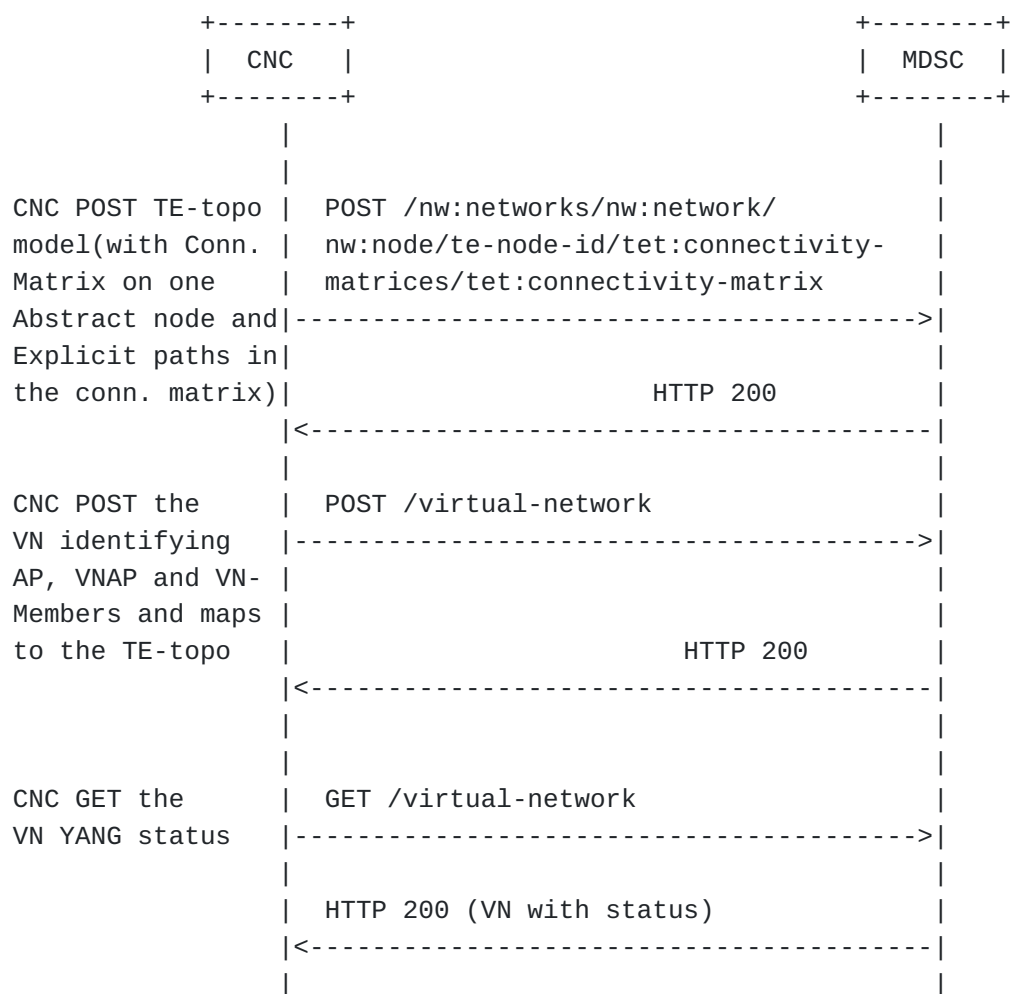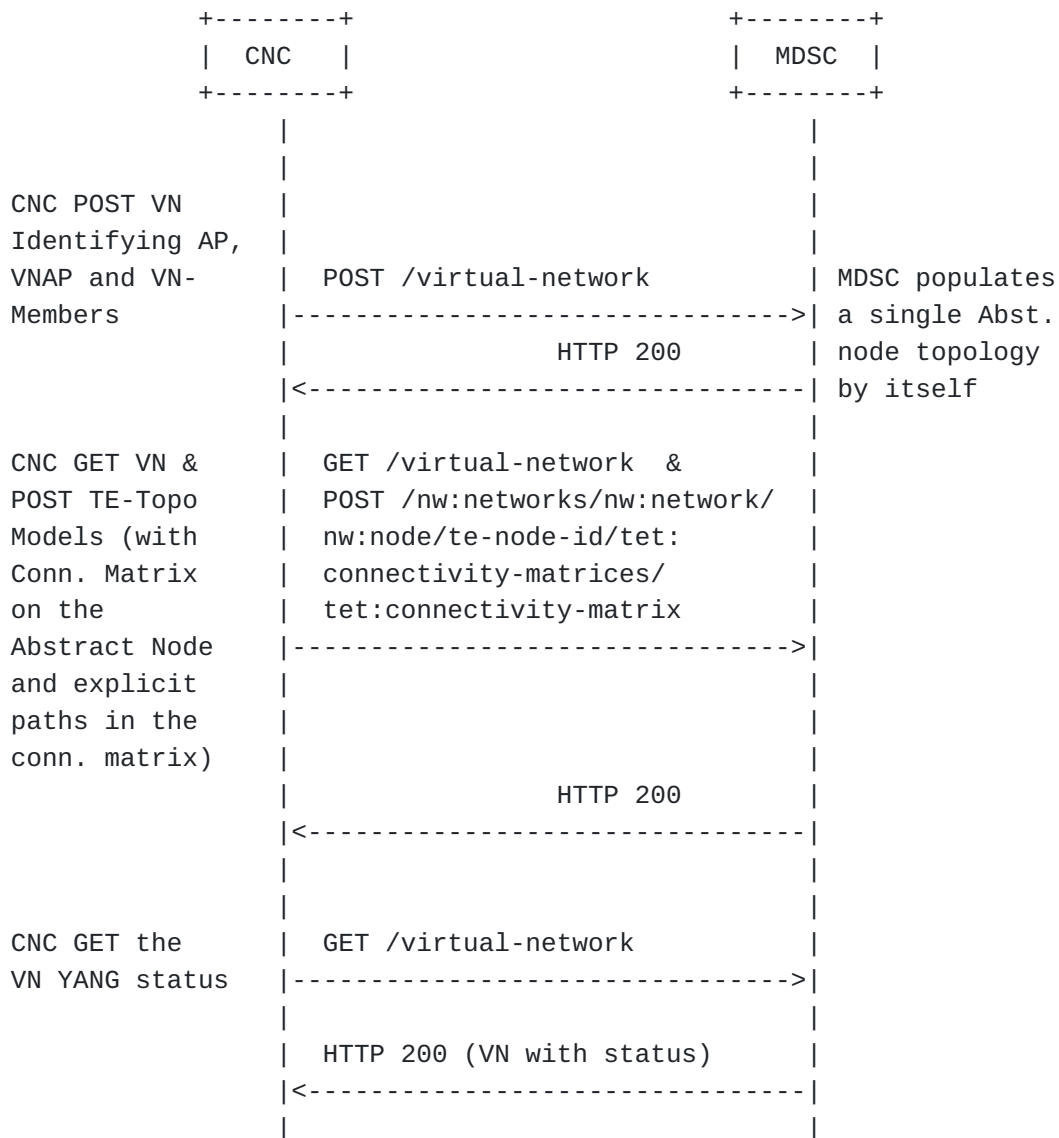
Case 2:

   On the other hand, if MDSC create the single abstract node topology
   based VN YANG posted by the CNC, the following diagram shows the
   message flow between CNC and MDSC to instantiate this VN using VN
   and TE-Topology Models.

```
            +--------+                    +--------+
            |  CNC   |                    |  MDSC  |
            +--------+                    +--------+
                 |                            |
                 |                            |
CNC POST VN      |                            |
Identifying AP,  |                            |
VNAP and VN-     | POST /virtual-network      | MDSC populates
Members          |--------------------------->| a single Abst.
                 |               HTTP 200      | node topology
                 |<---------------------------| by itself
                 |                            |
CNC GET VN &     |  GET /virtual-network  &   |
POST TE-Topo     |  POST /nw:networks/nw:network/ |
Models (with     |  nw:node/te-node-id/tet:   |
Conn. Matrix     |  connectivity-matrices/    |
on the           |  tet:connectivity-matrix   |
Abstract Node    |--------------------------->|
and explicit     |                            |
paths in the     |                            |
conn. matrix)    |                            |
                 |               HTTP 200      |
                 |<---------------------------|
                 |                            |
                 |                            |
CNC GET the      |  GET /virtual-network      |
VN YANG status   |--------------------------->|
                 |                            |
                 |  HTTP 200 (VN with status) |
                 |<---------------------------|
                 |                            |
```

Section 7 provides JSON examples for both VN model and TE-topology
Connectivity Matrix sub-model to illustrate how a VN can be created
by the CNC making use of the VN module as well as the TE-topology
Connectivity Matrix module.

3.2.1. VN and AP Usage

The customer access information may be known at the time of VN
creation. A shared logical AP identifier is used between the
customer and the operator to identify the access link between
Customer Edge (CE) and Provider Edge (PE) . This is described in
Section 6 of [RFC8453].

In some VN operations, the customer access may not be known at the
initial VN creation. The VN operation allow a creation of VN with
only PE identifier as well. The customer access information could be
added later.

To achieve this the 'ap' container has a leaf for 'pe' node that
allows AP to be created with PE information. The vn-member (and vn)
could use APs that only have PE information initially.

4. VN Model Usage

4.1. Customer view of VN

The VN-YANG model allows to define a customer view, and allows the
customer to communicate using the VN constructs as described in the
[RFC8454]. It also allows to group the set of edge-to-edge links
(i.e., VN members) under a common umbrella of VN. This allows the
customer to instantiate and view the VN as one entity, making it
easier for some customers to work on VN without worrying about the
details of the provider based YANG models.

This is similar to the benefits of having a separate YANG model for
the customer services as described in [RFC8309], which states that
service models do not make any assumption of how a service is
actually engineered and delivered for a customer.

4.2. Auto-creation of VN by MDSC

The VN could be configured at the MDSC explicitly by the CNC using
the VN YANG model. In some other cases, the VN is not explicitly
configured, but created automatically by the MDSC based on the
customer service model and local policy, even in these case the VN
YANG model can be used by the CNC to learn details of the underlying
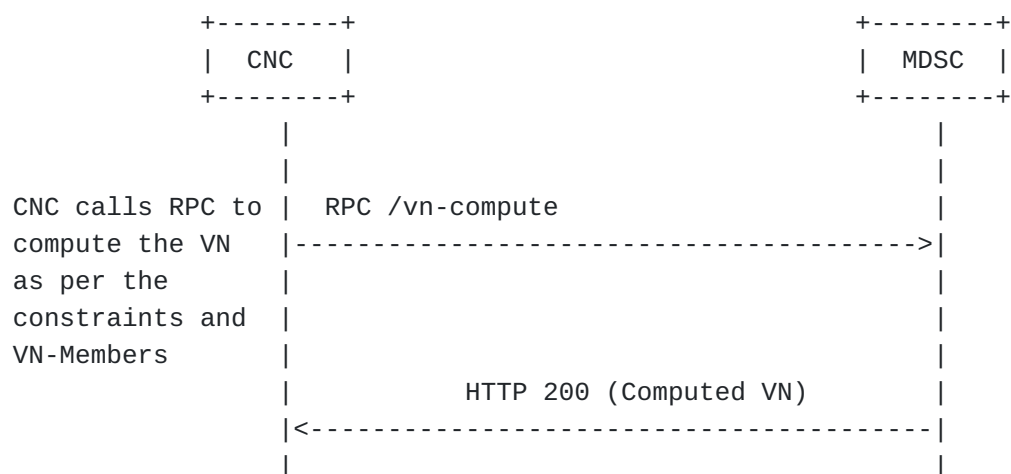VN created to meet the requirements of customer service model.

4.3. Innovative Services

4.3.1. VN Compute

VN Model supports VN compute (pre-instantiation mode) to view the
full VN as a single entity before instantiation. Achieving this via
path computation or "compute only" tunnel setup does not provide the
same functionality.

```
          +--------+                        +--------+
          |  CNC   |                        |  MDSC  |
          +--------+                        +--------+
              |                                  |
              |                                  |
CNC POST TE-topo |   POST /nw:networks/nw:network/       |
model(with Conn. |   nw:node/te-node-id/tet:connectivity- |
Matrix on one    |   matrices/tet:connectivity-matrix     |
Abstract node and|-------------------------------------->|
constraints in   |                                  |
the conn. matrix)|              HTTP 200            |
              |<--------------------------------------|
              |                                  |
              |                                  |
CNC calls RPC to |   RPC /vn-compute              |
compute the VN   |-------------------------------------->|
as per the       |                                  |
refered TE-Topo  |                                  |
              |                                  |
              |         HTTP 200 (Computed VN)       |
              |<--------------------------------------|
              |                                  |
```
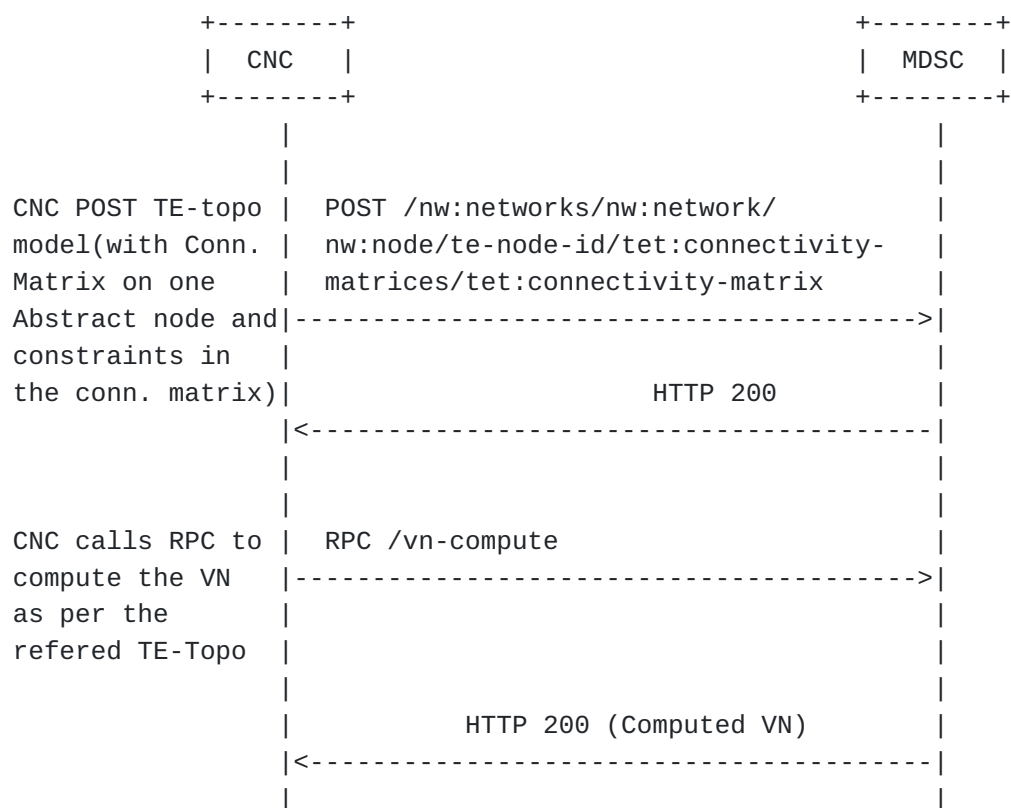
   The VN compute RPC allow you to optionally include the constraints
   and the optimization criteria at the VN as well as at the individual
   VN-member level. Thus, the RPC can be used independently to get the
   computed VN result without creating an abstract topology first.

```
          +--------+                        +--------+
          |  CNC   |                        |  MDSC  |
          +--------+                        +--------+
              |                                  |
              |                                  |
CNC calls RPC to |   RPC /vn-compute              |
compute the VN   |-------------------------------------->|
as per the       |                                  |
constraints and  |                                  |
VN-Members       |                                  |
              |         HTTP 200 (Computed VN)       |
              |<--------------------------------------|
              |                                  |
```

   In either case the output includes a reference to the single node
   abstract topology with each VN-member including a reference to the
   connectivity-matrix-id where the path properties could be found.

To achieve this the VN-compute RPC reuses the following common groupings:

  *te-types:generic-path-constraints: This is used optionally in the RPC input at the VN and/or VN-member level. The VN-member level overrides the VN-level data. This also overrides any constraints in the referred abstract node in the TE topology.

  *te-types:generic-path-optimization: This is used optionally in the RPC input at the VN and/or VN-member level. The VN-member level overrides the VN-level data. This also overrides any optimization in the referred abstract node in the TE topology.

  *vn-member: This identifies the VN member in both RPC input and output.

  *vn-policy: This is used optionally in the RPC input to apply any VN level policies.

When MDSC receives this RPC it computes the VN based on the input provided in the RPC call. This computation does not create a VN or reserve any resources in the system, it simply computes the resulting VN based on information at the MDSC or in coordination with the CNC. A single node abstract topology is used to convey the result of the each VN member as a reference to the connectivity-matrix-id. In case of error, the error information is included.

```
rpcs:
  +---x vn-compute
     +---w input
     |  +---w abstract-node?
     |  |        -> /nw:networks/network/node/tet:te-node-id
     |  +---w path-constraints
     |  |     ...
     |  +---w optimizations
     |  |     ...
     |  +---w vn-member-list* [vnm-id]
     |  |  +---w vnm-id                 vnm-id
     |  |  +---w src
     |  |  |  +---w src?          -> /access-point/ap/ap-id
     |  |  |  +---w src-vn-ap-id?  -> /access-point/ap/vn-ap/vn-ap-id
     |  |  |  +---w multi-src?     boolean {multi-src-dest}?
     |  |  +---w dest
     |  |  |  +---w dest?          -> /access-point/ap/ap-id
     |  |  |  +---w dest-vn-ap-id?  -> /access-point/ap/vn-ap/vn-ap-id
     |  |  |  +---w multi-dest?     boolean {multi-src-dest}?
     |  |  +---w connectivity-matrix-id?   leafref
     |  |  +--rw underlay
     |  |  +---w path-constraints
     |  |  |  |     ...
     |  |  +---w optimizations
     |  |           ...
     |  +---w vn-level-diversity?   te-types:te-path-disjointness
     +--ro output
        +--ro abstract-node?
        |        -> /nw:networks/network/node/tet:te-node-id
        +--ro vn-member-list* [vnm-id]
           +--ro vnm-id                 vnm-id
           +--ro src
           |  +--ro src?          -> /access-point/ap/ap-id
           |  +--ro src-vn-ap-id?  -> /access-point/ap/vn-ap/vn-ap-id
           |  +--ro multi-src?     boolean {multi-src-dest}?
           +--ro dest
           |  +--ro dest?          -> /access-point/ap/ap-id
           |  +--ro dest-vn-ap-id?  -> /access-point/ap/vn-ap/vn-ap-id
           |  +--ro multi-dest?     boolean {multi-src-dest}?
           +--ro connectivity-matrix-id?   leafref
           +--rw underlay
           +--ro if-selected?            boolean
           |        {multi-src-dest}?
           +--ro compute-status?         vn-compute-status
           +--ro error-info
              +--ro error-description?   string
              +--ro error-timestamp?     yang:date-and-time
              +--ro error-reason?        identityref
```

4.3.2. Multi-sources and Multi-destinations

In creating a virtual network, the list of sources or destinations
or both may not be pre-determined by the customer. For instance, for
a given source, there may be a list of multiple-destinations to
which the optimal destination may be chosen depending on the network
resource situations. Likewise, for a given destination, there may
also be multiple-sources from which the optimal source may be
chosen. In some cases, there may be a pool of multiple sources and
destinations from which the optimal source-destination may be
chosen. The following YANG module is shown for describing source
container and destination container. The following YANG tree shows
how to model multi-sources and multi-destinations.

```
+--rw virtual-network
   +--rw vn* [vn-id]
      +--rw vn-id                   vn-id
      +--rw vn-topology-id?        te-types:te-topology-id
      +--rw abstract-node?
      |         -> /nw:networks/network/node/tet:te-node-id
      +--rw vn-member* [vnm-id]
      |  +--rw vnm-id                    vnm-id
      |  +--rw src
      |  |  +--rw src?           -> /access-point/ap/ap-id
      |  |  +--rw src-vn-ap-id?  -> /access-point/ap/vn-ap/vn-ap-id
      |  |  +--rw multi-src?     boolean {multi-src-dest}?
      |  +--rw dest
      |  |  +--rw dest?          -> /access-point/ap/ap-id
      |  |  +--rw dest-vn-ap-id? -> /access-point/ap/vn-ap/vn-ap-id
      |  |  +--rw multi-dest?     boolean {multi-src-dest}?
      |  +--rw connectivity-matrix-id?   leafref
      |  +--rw underlay
      |  +--ro oper-status?              te-types:te-oper-status
      +--ro if-selected?        boolean {multi-src-dest}?
      +--rw admin-status?       te-types:te-admin-status
      +--ro oper-status?        te-types:te-oper-status
      +--rw vn-level-diversity?   te-types:te-path-disjointness
```

4.3.3. Others

The VN YANG model can be easily augmented to support the mapping of
VN to the Services such as L3SM and L2SM as described in [I-D.ietf-
teas-te-service-mapping-yang].

The VN YANG model can be extended to support telemetry, performance
monitoring and network autonomics as described in [I-D.ietf-teas-
actn-pm-telemetry-autonomics].

Note that the YANG model is tightly coupled with the TE Topology
model [RFC8795]. Any underlay technology not supported by [RFC8795]
is also not supported by this model. The model does include an empty
container called "underlay" that can be augmented. For example the
SR-policy information can be augmented for the SR underlay by a
future model.

4.3.4. Summary

This section summarizes the innovative service features of the VN
YANG.

   *Maintenance of AP and VNAP along with VN

   *VN construct to group of edge-to-edge links

   *VN Compute (pre-instantiate)

   *Multi-Source / Multi-Destination

   *Ability to support various VN and VNS Types

      -VN Type 1: Customer configures the VN as a set of VN Members.
       No other details need to be set by customer, making for a
       simplified operations for the customer.

      -VN Type 2: Along with VN Members, the customer could also
       provide an abstract topology, this topology is provided by the
       Abstract TE Topology YANG Model.

5. VN YANG Model (Tree Structure)

```
module: ietf-vn
  +--rw access-point
  |  +--rw ap* [ap-id]
  |     +--rw ap-id              ap-id
  |     +--rw pe?
  |     |          -> /nw:networks/network/node/tet:te-node-id
  |     +--rw max-bandwidth?    te-types:te-bandwidth
  |     +--rw avl-bandwidth?    te-types:te-bandwidth
  |     +--rw vn-ap* [vn-ap-id]
  |        +--rw vn-ap-id          ap-id
  |        +--rw vn?               -> /virtual-network/vn/vn-id
  |        +--rw abstract-node?
  |        |          -> /nw:networks/network/node/tet:te-node-id
  |        +--rw ltp?              leafref
  |        +--ro max-bandwidth?    te-types:te-bandwidth
  +--rw virtual-network
     +--rw vn* [vn-id]
        +--rw vn-id               vn-id
        +--rw vn-topology-id?      te-types:te-topology-id
        +--rw abstract-node?
        |          -> /nw:networks/network/node/tet:te-node-id
        +--rw vn-member* [vnm-id]
        |  +--rw vnm-id                   vnm-id
        |  +--rw src
        |  |  +--rw src?             -> /access-point/ap/ap-id
        |  |  +--rw src-vn-ap-id?
        |  |  |          -> /access-point/ap/vn-ap/vn-ap-id
        |  |  +--rw multi-src?       boolean {multi-src-dest}?
        |  +--rw dest
        |  |  +--rw dest?            -> /access-point/ap/ap-id
        |  |  +--rw dest-vn-ap-id?
        |  |  |          -> /access-point/ap/vn-ap/vn-ap-id
        |  |  +--rw multi-dest?      boolean {multi-src-dest}?
        |  +--rw connectivity-matrix-id?   leafref
        |  +--rw underlay
        |  +--ro oper-status?             te-types:te-oper-status
        +--ro if-selected?         boolean {multi-src-dest}?
        +--rw admin-status?        te-types:te-admin-status
        +--ro oper-status?         te-types:te-oper-status
        +--rw vn-level-diversity?  te-types:te-path-disjointness

  rpcs:
    +---x vn-compute
       +---w input
       |  +---w abstract-node?
       |  |          -> /nw:networks/network/node/tet:te-node-id
       |  +---w path-constraints
       |  |  +---w te-bandwidth
       |  |  |  +---w (technology)?
```

```
|  |  |        ...
|  |  +---w link-protection?          identityref
|  |  +---w setup-priority?           uint8
|  |  +---w hold-priority?            uint8
|  |  +---w signaling-type?           identityref
|  |  +---w path-metric-bounds
|  |  |  +---w path-metric-bound* [metric-type]
|  |  |        ...
|  |  +---w path-affinities-values
|  |  |  +---w path-affinities-value* [usage]
|  |  |        ...
|  |  +---w path-affinity-names
|  |  |  +---w path-affinity-name* [usage]
|  |  |        ...
|  |  +---w path-srlgs-lists
|  |  |  +---w path-srlgs-list* [usage]
|  |  |        ...
|  |  +---w path-srlgs-names
|  |  |  +---w path-srlgs-name* [usage]
|  |  |        ...
|  |  +---w disjointness?            te-path-disjointness
|  +---w cos?               te-types:te-ds-class
|  +---w optimizations
|  |  +---w (algorithm)?
|  |     +--:(metric) {path-optimization-metric}?
|  |     |     ...
|  |     +--:(objective-function)
|  |           {path-optimization-objective-function}?
|  |           ...
|  +---w vn-member-list* [vnm-id]
|  |  +---w vnm-id                 vnm-id
|  |  +---w src
|  |  |  +---w src?          -> /access-point/ap/ap-id
|  |  |  +---w src-vn-ap-id?
|  |  |  |     -> /access-point/ap/vn-ap/vn-ap-id
|  |  |  +---w multi-src?     boolean {multi-src-dest}?
|  |  +---w dest
|  |  |  +---w dest?          -> /access-point/ap/ap-id
|  |  |  +---w dest-vn-ap-id?
|  |  |  |     -> /access-point/ap/vn-ap/vn-ap-id
|  |  |  +---w multi-dest?     boolean {multi-src-dest}?
|  |  +---w connectivity-matrix-id?   leafref
|  |  +---w underlay
|  |  +---w path-constraints
|  |  |  +---w te-bandwidth
|  |  |  |     ...
|  |  |  +---w link-protection?          identityref
|  |  |  +---w setup-priority?           uint8
|  |  |  +---w hold-priority?            uint8
```

```
|  |  |  +---w signaling-type?           identityref
|  |  |  +---w path-metric-bounds
|  |  |  |     ...
|  |  |  +---w path-affinities-values
|  |  |  |     ...
|  |  |  +---w path-affinity-names
|  |  |  |     ...
|  |  |  +---w path-srlgs-lists
|  |  |  |     ...
|  |  |  +---w path-srlgs-names
|  |  |  |     ...
|  |  |  +---w disjointness?           te-path-disjointness
|  |  +---w cos?                    te-types:te-ds-class
|  |  +---w optimizations
|  |     +---w (algorithm)?
|  |        ...
|  +---w vn-level-diversity?   te-types:te-path-disjointness
+--ro output
   +--ro abstract-node?
   |        -> /nw:networks/network/node/tet:te-node-id
   +--ro vn-member-list* [vnm-id]
      +--ro vnm-id                   vnm-id
      +--ro src
      |  +--ro src?           -> /access-point/ap/ap-id
      |  +--ro src-vn-ap-id?
      |  |        -> /access-point/ap/vn-ap/vn-ap-id
      |  +--ro multi-src?     boolean {multi-src-dest}?
      +--ro dest
      |  +--ro dest?          -> /access-point/ap/ap-id
      |  +--ro dest-vn-ap-id?
      |  |        -> /access-point/ap/vn-ap/vn-ap-id
      |  +--ro multi-dest?     boolean {multi-src-dest}?
      +--ro connectivity-matrix-id?   leafref
      +--ro underlay
      +--ro if-selected?           boolean
      |        {multi-src-dest}?
      +--ro compute-status?        vn-compute-status
      +--ro error-info
         +--ro error-description?   string
         +--ro error-timestamp?     yang:date-and-time
         +--ro error-reason?        identityref
```

6. VN YANG Model

The YANG model is as follows:

```
<CODE BEGINS> file "ietf-vn@2021-10-23.yang"

module ietf-vn {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-vn";
  prefix vn;

  /* Import network */

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-network {
    prefix nw;
    reference
      "RFC 8345: A YANG Data Model for Network Topologies";
  }

  /* Import network topology */

  import ietf-network-topology {
    prefix nt;
    reference
      "RFC 8345: A YANG Data Model for Network Topologies";
  }

  /* Import TE Common types */

  import ietf-te-types {
    prefix te-types;
    reference
      "RFC 8776: Common YANG Data Types for Traffic Engineering";
  }

  /* Import TE Topology */

  import ietf-te-topology {
    prefix tet;
    reference
      "RFC 8795: YANG Data Model for Traffic Engineering (TE)
       Topologies";
  }

  organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS)
     Working Group";
  contact
    "WG Web:  <https://tools.ietf.org/wg/teas/>
```

```
   WG List:  <mailto:teas@ietf.org>
   Editor: Young Lee <younglee.tx@gmail.com>
         : Dhruv Dhody <dhruv.ietf@gmail.com>";
description
  "This module contains a YANG module for the VN. It describes a
   VN operation module that takes place in the context of the
   CNC-MDSC Interface (CMI) of the ACTN architecture where the
   CNC is the actor of a VN Instantiation/modification/deletion
   as per RFC 8453.

   Copyright (c) 2021 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject to
   the license terms contained in, the Simplified BSD License set
   forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see the
   RFC itself for full legal notices.

   The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
   NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
   'MAY', and 'OPTIONAL' in this document are to be interpreted as
   described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here.";

revision 2021-10-23 {
  description
    "initial version.";
  reference
    "RFC XXXX: A YANG Data Model for VN Operation";
}

/* Features */

feature multi-src-dest {
  description
    "Support for selection of one src or destination
     among multiple.";
  reference
    "RFC 8453: Framework for Abstraction and Control of TE
     Networks (ACTN)";
}

/* Typedef */

typedef vn-id {
```

```
    type string;
    description
      "Defines a type of Virtual Network (VN) identifier.";
}

typedef ap-id {
  type string;
  description
    "Defines a type of Access Point (AP) identifier.";
}

typedef vnm-id {
  type string;
  description
    "Defines a type of VN member identifier.";
}

typedef vn-compute-status {
  type te-types:te-common-status;
  description
    "Defines a type representing the VN compute status. Note
     that all status apart from up and down are considered as
     unknown.";
}

/* identities */

identity vn-computation-error-reason {
  description
    "Base identity for VN computation error reasons.";
}

identity vn-computation-error-not-ready {
  base vn-computation-error-reason;
  description
    "VN computation has failed because the MDSC is not
     ready";
}

identity vn-computation-error-no-cnc {
  base vn-computation-error-reason;
  description
    "VN computation has failed because one or more dependent
     CNC are unavailable.";
}

identity vn-computation-error-no-resource {
  base vn-computation-error-reason;
  description
    "VN computation has failed because there is no
```

```
      available resource in one or more domains.";
  }

  identity vn-computation-error-path-not-found {
    base vn-computation-error-reason;
    description
      "VN computation failed as no path found.";
  }

  identity vn-computation-ap-unknown {
    base vn-computation-error-reason;
    description
      "VN computation failed as source or destination AP not
       known.";
  }

  /* Groupings */

  grouping vn-ap {
    description
      "VNAP related information";
    leaf vn-ap-id {
      type ap-id;
      description
        "A unique identifier for the referred VNAP";
    }
    leaf vn {
      type leafref {
        path "/virtual-network/vn/vn-id";
      }
      description
        "A reference to the VN";
    }
    leaf abstract-node {
      type leafref {
        path "/nw:networks/nw:network/nw:node/tet:te-node-id";
      }
      description
        "A reference to the abstract node in TE Topology that
         represent the VN";
    }
    leaf ltp {
      type leafref {
        path "/nw:networks/nw:network/nw:node/"
            + "nt:termination-point/tet:te-tp-id";
      }
      description
        "A reference to Link Termination Point (LTP) in the
         TE-topology";
```

```
        reference
          "RFC 8795: YANG Data Model for Traffic Engineering (TE)
           Topologies";
      }
    leaf max-bandwidth {
      type te-types:te-bandwidth;
      config false;
      description
        "The max bandwidth of the VNAP";
    }
    reference
      "RFC 8453: Framework for Abstraction and Control of TE
       Networks (ACTN), Section 6";
} //vn-ap

grouping access-point {
    description
      "AP related information";
    leaf ap-id {
      type ap-id;
      description
        "A unique identifier for the referred access point";
    }
    leaf pe {
      type leafref {
        path "/nw:networks/nw:network/nw:node/tet:te-node-id";
      }
      description
        "A reference to the PE node in the native TE Topology";
    }
    leaf max-bandwidth {
      type te-types:te-bandwidth;
      description
        "The max bandwidth of the AP";
    }
    leaf avl-bandwidth {
      type te-types:te-bandwidth;
      description
        "The available bandwidth of the AP";
    }
    /*add details and any other properties of AP,
    not associated by a VN
    CE port, PE port etc.
     */
    list vn-ap {
      key "vn-ap-id";
      uses vn-ap;
      description
        "List of VNAP in this AP";
```

```
      }
      reference
        "RFC 8453: Framework for Abstraction and Control of TE
         Networks (ACTN), Section 6";
  } //access-point

  grouping vn-member {
    description
      "The vn-member is described by this grouping";
    leaf vnm-id {
      type vnm-id;
      description
        "A vn-member identifier";
    }
    container src {
      description
        "The source of VN Member";
      leaf src {
        type leafref {
          path "/access-point/ap/ap-id";
        }
        description
          "A reference to source AP";
      }
      leaf src-vn-ap-id {
        type leafref {
          path "/access-point/ap/vn-ap/vn-ap-id";
        }
        description
          "A reference to source VNAP";
      }
      leaf multi-src {
        if-feature "multi-src-dest";
        type boolean;
        default "false";
        description
          "Is the source part of multi-source, where
           only one of the source is enabled";
      }
    }
    container dest {
      description
        "the destination of VN Member";
      leaf dest {
        type leafref {
          path "/access-point/ap/ap-id";
        }
        description
          "A reference to destination AP";
```

```
        }
        leaf dest-vn-ap-id {
          type leafref {
            path "/access-point/ap/vn-ap/vn-ap-id";
          }
          description
            "A reference to dest VNAP";
        }
        leaf multi-dest {
          if-feature "multi-src-dest";
          type boolean;
          default "false";
          description
            "Is destination part of multi-destination, where only one
             of the destination is enabled";
        }
      }
      leaf connectivity-matrix-id {
        type leafref {
          path "/nw:networks/nw:network/nw:node/tet:te/"
             + "tet:te-node-attributes/"
             + "tet:connectivity-matrices/"
             + "tet:connectivity-matrix/tet:id";
        }
        description
          "A reference to connectivity-matrix";
        reference
          "RFC 8795: YANG Data Model for Traffic Engineering (TE)
           Topologies";
      }
      container underlay {
        description
          "An empty container that can be augmented with underlay
           technology information not supported by RFC 8795 (for
           example - Segement Routing (SR). ";
      }
      reference
        "RFC 8454: Information Model for Abstraction and Control of TE
         Networks (ACTN)";
    } //vn-member

    grouping vn-policy {
      description
        "policy for VN-level diverisity";
      leaf vn-level-diversity {
        type te-types:te-path-disjointness;
        description
          "The type of disjointness on the VN level (i.e., across all
           VN members)";
```

```
    }
  }

  /* Configuration data nodes */

  container access-point {
    description
      "AP configurations";
    list ap {
      key "ap-id";
      description
        "access-point identifier";
      uses access-point {
        description
          "The access-point information";
      }
    }
    reference
      "RFC 8453: Framework for Abstraction and Control of TE
       Networks (ACTN), Section 6";
  }
  container virtual-network {
    description
      "VN configurations";
    list vn {
      key "vn-id";
      description
        "A virtual network is identified by a vn-id";
      leaf vn-id {
        type vn-id;
        description
          "A unique VN identifier";
      }
      leaf vn-topology-id {
        type te-types:te-topology-id;
        description
          "An optional identifier to the TE Topology Model where the
           abstract nodes and links of the Topology can be found for
           Type 2 VNS";
      }
      leaf abstract-node {
        type leafref {
          path "/nw:networks/nw:network/nw:node/tet:te-node-id";
        }
        description
          "A reference to the abstract node in TE Topology";
      }
      list vn-member {
        key "vnm-id";
```

```
      description
        "List of vn-members in a VN";
      uses vn-member;
      leaf oper-status {
        type te-types:te-oper-status;
        config false;
        description
          "The vn-member operational state.";
      }
    }
    leaf if-selected {
      if-feature "multi-src-dest";
      type boolean;
      default "false";
      config false;
      description
        "Is the vn-member is selected among the multi-src/dest
         options";
    }
    leaf admin-status {
      type te-types:te-admin-status;
      default "up";
      description
        "VN administrative state.";
    }
    leaf oper-status {
      type te-types:te-oper-status;
      config false;
      description
        "VN operational state.";
    }
    uses vn-policy;
  } //vn
  reference
    "RFC 8453: Framework for Abstraction and Control of TE
     Networks (ACTN)";
} //vn

/* RPC */

rpc vn-compute {
  description
    "The VN computation without actual instantiation. This is
     used by the CNC to get the VN results without actually
     creating it in the network.

     The input could include a reference to the single node
     abstract topology. It could optionally also include
     constraints and optimization criteria. The computation
```

```
      is done based on the list of VN-members.

      The output includes a reference to the single node
      abstract topology with each VN-member including a
      reference to the connectivity-matrix-id where the
      path properties could be found. Error information is
      also included.";
  input {
    leaf abstract-node {
      type leafref {
        path "/nw:networks/nw:network/nw:node/tet:te-node-id";
      }
      description
        "A reference to the abstract node in TE Topology";
    }
    uses te-types:generic-path-constraints;
    leaf cos {
      type te-types:te-ds-class;
      description
        "The class of service";
    }
    uses te-types:generic-path-optimization;
    list vn-member-list {
      key "vnm-id";
      description
        "List of VN-members in a VN";
      uses vn-member;
      uses te-types:generic-path-constraints;
      leaf cos {
        type te-types:te-ds-class;
        description
          "The class of service";
      }
      uses te-types:generic-path-optimization;
    }
    uses vn-policy;
  }
  output {
    leaf abstract-node {
      type leafref {
        path "/nw:networks/nw:network/nw:node/tet:te-node-id";
      }
      description
        "A reference to the abstract node in TE Topology";
    }
    list vn-member-list {
      key "vnm-id";
      description
        "List of VN-members in a VN";
```

```
      uses vn-member;
      leaf if-selected {
        if-feature "multi-src-dest";
        type boolean;
        default "false";
        description
          "Is the vn-member is selected among the multi-src/dest
           options";
        reference
          "RFC 8453: Framework for Abstraction and Control of TE
           Networks (ACTN), Section 7";
      }
      leaf compute-status {
        type vn-compute-status;
        description
          "The VN-member compute state.";
      }
      container error-info {
        description
          "Error information related to the VN member";
        leaf error-description {
          type string;
          description
            "Textual representation of the error occurred during
             VN compute.";
        }
        leaf error-timestamp {
          type yang:date-and-time;
          description
            "Timestamp of the attempt.";
        }
        leaf error-reason {
          type identityref {
            base vn-computation-error-reason;
          }
          description
            "Reason for the VN computation error.";
        }
      }
    }
  }
} //vn-compute

}


<CODE ENDS>
```

7. JSON Example

This section provides json implementation examples as to how VN YANG model and TE topology model are used together to instantiate virtual networks.

The example in this section includes following VN

  *VN1 (Type 1): Which maps to the single node topology abstract1
   (node D1) and consist of VN Members 104 (L1 to L4), 107 (L1 to
   L7), 204 (L2 to L4), 308 (L3 to L8) and 108 (L1 to L8). We also
   show how disjointness (node, link, srlg) is supported in the
   example on the global level (i.e., connectivity matrices level).

  *VN2 (Type 2): Which maps to the single node topology abstract2
   (node D2), this topology has an underlay topology (absolute) (see
   figure in section 3.2). This VN has a single VN member 105 (L1 to
   L5) and an underlay path (S4 and S7) has been set in the
   connectivity matrix of abstract2 topology;

  *VN3 (Type 1): This VN has a multi-source, multi-destination
   feature enable for VN Member 104 (L1 to L4)/107 (L1 to L7)
   {multi-src} and VN Member 204 (L2 to L4)/304 (L3 to L4) {multi-
   dest} usecase. The selected VN-member is known via the field "if-
   selected" and the corresponding connectivity-matrix-id.

Note that the VN YANG model also include the AP and VNAP which shows
various VN using the same AP.

7.1. VN JSON

```json
{
    "access-point":{
        "ap": [
            {
                "ap-id": "101",
                "vn-ap": [
                    {
                        "vn-ap-id": "10101",
                        "vn": "1",
                        "abstract-node": "D1",
                        "ltp": "1-0-1"
                    },
                    {
                        "vn-ap-id": "10102",
                        "vn": "2",
                        "abstract-node": "D2",
                        "ltp": "1-0-1"
                    },
                    {
                        "vn-ap-id": "10103",
                        "vn": "3",
                        "abstract-node": "D3",
                        "ltp": "1-0-1"
                    },
                ]
            },
            {
                "ap-id": "202",
                "vn-ap": [
                    {
                        "vn-ap-id": "20201",
                        "vn": "1",
                        "abstract-node": "D1",
                        "ltp": "2-0-2"
                    }
                ]
            },
            {
                "ap-id": "303",
                "vn-ap": [
                    {
                        "vn-ap-id": "30301",
                        "vn": "1",
                        "abstract-node": "D1",
                        "ltp": "3-0-3"
                    },
                    {
                        "vn-ap-id": "30303",
                        "vn": "3",
```

```
                    "abstract-node": "D3",
                    "ltp": "3-0-3"
                }
            ]
        },
        {
            "ap-id": "440",
            "vn-ap": [
                {
                    "vn-ap-id": "44001",
                    "vn": "1",
                    "abstract-node": "D1",
                    "ltp": "4-4-0"
                }
            ]
        },
        {
            "ap-id": "550",
            "vn-ap": [
                {
                    "vn-ap-id": "55002",
                    "vn": "2",
                    "abstract-node": "D2",
                    "ltp": "5-5-0"
                }
            ]
        },
        {
            "ap-id": "770",
            "vn-ap": [
                {
                    "vn-ap-id": "77001",
                    "vn": "1",
                    "abstract-node": "D1",
                    "ltp": "7-7-0"
                },
                {
                    "vn-ap-id": "77003",
                    "vn": "3",
                    "abstract-node": "D3",
                    "ltp": "7-7-0"
                }
            ]
        },
        {
            "ap-id": "880",
            "vn-ap": [
                {
                    "vn-ap-id": "88001",
```

```json
                    "vn": "1",
                    "abstract-node": "D1",
                    "ltp": "8-8-0"
                },
                {

                    "vn-ap-id": "88003",
                    "vn": "3",
                    "abstract-node": "D3",
                    "ltp": "8-8-0"
                }
            ]
        }
    ]
},
"virtual-network":{
    "vn": [
        {
            "vn-id": "1",
            "vn-topology-id": "te-topology:abstract1",
            "abstract-node": "D1",
            "vn-member": [
                {
                    "vnm-id": "104",
                    "src": {
                        "src": "101",
                        "src-vn-ap-id": "10101",
                    },
                    "dest": {
                        "dest": "440",
                        "dest-vn-ap-id": "44001",
                    },
                    "connectivity-matrix-id": 104
                },
                {
                    "vnm-id": "107",
                    "src": {
                        "src": "101",
                        "src-vn-ap-id": "10101",
                    },
                    "dest": {
                        "dest": "770",
                        "dest-vn-ap-id": "77001",
                    },
                    "connectivity-matrix-id": 107
                },
                {
                    "vnm-id": "204",
                    "src": {
                        "src": "202",
```

```json
                "dest-vn-ap-id": "20401",
            },
            "dest": {
                "dest": "440",
                "dest-vn-ap-id": "44001",
            },
            "connectivity-matrix-id": 204
        },
        {
            "vnm-id": "308",
            "src": {
                "src": "303",
                "src-vn-ap-id": "30301",
            },
            "dest": {
                "dest": "880",
                "src-vn-ap-id": "88001",
            },
            "connectivity-matrix-id": 308
        },
        {
            "vnm-id": "108",
            "src": {
                "src": "101",
                "src-vn-ap-id": "10101",
            },
            "dest": {
                "dest": "880",
                "dest-vn-ap-id": "88001",
            },
            "connectivity-matrix-id": "108"
        }
    ]
},
{
    "vn-id": "2",
    "vn-topology-id": "te-topology:abstract2",
    "abstract-node": "D2",
    "vn-member": [
        {
            "vnm-id": "105",
            "src": {
                "src": "101",
                "src-vn-ap-id": "10102",
            },
            "dest": {
                "dest": "550",
                "dest-vn-ap-id": "55002",
            },
```

```
                    "connectivity-matrix-id": 105
                }
            ]
        },
        {
            "vn-id": "3",
            "vn-topology-id": "te-topology:abstract3",
            "abstract-node": "D3",
            "vn-member": [
                {
                    "vnm-id": "104",
                    "src": {
                        "src": "101",
                    },
                    "dest": {
                        "dest": "440",
                        "multi-dest": true
                    }
                },
                {
                    "vnm-id": "107",
                    "src": {
                        "src": "101",
                        "src-vn-ap-id": "10103",
                    },
                    "dest": {
                        "dest": "770",
                        "dest-vn-ap-id": "77003",
                        "multi-dest": true
                    },
                    "connectivity-matrix-id": 107,
                    "if-selected":true,
                },
                {
                    "vnm-id": "204",
                    "src": {
                        "src": "202",
                        "multi-src": true,
                    },
                    "dest": {
                        "dest": "440",
                    },
                },
                {
                    "vnm-id": "304",
                    "src": {
                        "src": "303",
                        "src-vn-ap-id": "30303",
                        "multi-src": true,
```

```
                },
                "dest": {
                    "dest": "440",
                    "src-vn-ap-id": "44003",
                },
                "connectivity-matrix-id": 304,
                "if-selected":true,
            },
        ]
    },

    ]
}

}
}
```

7.2. TE-topology JSON

```json
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {}
        },
        "network-id": "abstract1",
        "te-topology-identifier":{
          "provider-id": 201,
          "client-id": 600,
          "topology-id": "te-topology:abstract1"
        },
        "node": [
          {
            "node-id": "D1",
            "te-node-id": "2.0.1.1",
            "te": {
              "te-node-attributes": {
                "domain-id" : 1,
                "is-abstract": [null],
                "connectivity-matrices": {
                  "is-allowed": true,
                  "path-constraints": {
                    "te-bandwidth": {
                      "generic": "0x1p10",
                    },
                    "disjointness": "node link srlg",
                  },
                  "connectivity-matrix": [
                    {
                      "id": 104,
                      "from": {
                        "tp-ref":"1-0-1",
                      },
                      "to": {
                        "tp-ref":"4-4-0",
                      },
                    },
                    {
                      "id": 107,
                      "from": {
                        "tp-ref":"1-0-1",
                      },
                      "to": {
                        "tp-ref":"7-7-0",
                      },
                    },
```

```
            {
              "id": 204,
              "from": {
                "tp-ref":"2-0-2",
              },
              "to": {
                "tp-ref":"4-4-0",
              },
            },
            {
              "id": 308,
              "from": {
                "tp-ref":"3-0-3",
              },
              "to": {
                "tp-ref":"8-8-0",
              },
            },
            {
              "id": 108,
              "from": {
                "tp-ref":"1-0-1",
              },
              "to": {
                "tp-ref":"8-8-0",
              },
            },
          ]
        }
      },
      "tunnel-termination-point": [
        {
          "name": "1-0-1",
          "tunnel-tp-id": 10001,
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk",
        },
        {
          "name": "1-1-0",
          "tunnel-tp-id": 10100,
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk",
        },
        {
          "name": "2-0-2",
          "tunnel-tp-id": 20002,
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk",
        },
```

```json
  {
    "name": "2-2-0",
    "tunnel-tp-id": 20200,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "3-0-3",
    "tunnel-tp-id": 30003,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "3-3-0",
    "tunnel-tp-id": 30300,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "4-0-4",
    "tunnel-tp-id": 40004,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "4-4-0",
    "tunnel-tp-id": 40400,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "5-0-5",
    "tunnel-tp-id": 50005,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "5-5-0",
    "tunnel-tp-id": 50500,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
    "name": "6-0-6",
    "tunnel-tp-id": 60006,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
  },
  {
```

```
                "name": "6-6-0",
                "tunnel-tp-id": 60600,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk",
            },
            {
                "name": "7-0-7",
                "tunnel-tp-id": 70007,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk",
            },
            {
                "name": "7-7-0",
                "tunnel-tp-id": 70700,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk",
            },
            {
                "name": "8-0-8",
                "tunnel-tp-id": 80008,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk",
            },
            {
                "name": "8-8-0",
                "tunnel-tp-id": 80800,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk",
            },
          ],
        },
      },
    ],
  },
  {
    "network-types": {
      "te-topology": {}
    },
    "network-id": "abstract2",
    "te-topology-identifier":{
      "provider-id": 201,
      "client-id": 600,
      "topology-id": "te-topology:abstract2"
    },
    "node": [
    {
      "node-id": "D2",
      "te-node-id": "2.0.1.2",
      "te": {
```

```json
"te-node-attributes": {
  "domain-id" : 1,
  "is-abstract": [null],
  "connectivity-matrices": {
    "is-allowed": true,
    "underlay": {
      "enabled": true
    },
    "path-constraints": {
      "te-bandwidth": {
        "generic": "0x1p10",
      },
    },
    "optimizations": {
      "objective-function": {
          "objective-function-type":
            "of-maximize-residual-bandwidth"
      },
    },
    "connectivity-matrix": [
      {
        "id": 105,
        "from": {
            "tp-ref": "1-0-1",
        },
        "to": {
            "tp-ref": "5-5-0",
        },
        "underlay": {
          "enabled": true,
          "primary-path": {
            "network-ref": "absolute",
            "path-element": [
              {
                "path-element-id": 1,
                "numbered-node-hop": {
                  "node-id": "4.4.4.4",
                  "hop-type": "strict"
                }
              },
              {
                "path-element-id": 2,
                "numbered-hop": {
                  "node-id": "7.7.7.7",
                  "hop-type": "strict"
                }
              }
            ]
          }
```

```
          }
        }
      ]
    }
  },
  "tunnel-termination-point": [
    {
      "name": "1-0-1",
      "tunnel-tp-id": 10001,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "1-1-0",
      "tunnel-tp-id": 10100,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "2-0-2",
      "tunnel-tp-id": 20002,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "2-2-0",
      "tunnel-tp-id": 20200,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "3-0-3",
      "tunnel-tp-id": 30003,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "3-3-0",
      "tunnel-tp-id": 30300,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "4-0-4",
      "tunnel-tp-id": 40004,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
```

```json
    "name": "4-4-0",
    "tunnel-tp-id": 40400,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "5-0-5",
    "tunnel-tp-id": 50005,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "5-5-0",
    "tunnel-tp-id": 50500,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
  "name": "6-0-6",
  "tunnel-tp-id": 60006,
  "switching-capability": "switching-otn",
  "encoding": "lsp-encoding-oduk",
},
{
    "name": "6-6-0",
    "tunnel-tp-id": 60600,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "7-0-7",
    "tunnel-tp-id": 70007,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "7-7-0",
    "tunnel-tp-id": 70700,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "8-0-8",
    "tunnel-tp-id": 80008,
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk",
},
{
    "name": "8-8-0",
```

```json
                "tunnel-tp-id": 80800,
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            },
        ],
        },
    },
    {
      "network-types": {
          "te-topology": {}
      },
      "network-id": "abstract3",
      "te-topology-identifier":{
          "provider-id": 201,
          "client-id": 600,
          "topology-id": "te-topology:abstract3"
      },
      "node": [
        {
          "node-id": "D3",
          "te-node-id": "3.0.1.1",
          "te": {
            "te-node-attributes": {
              "domain-id" : 3,
              "is-abstract": [null],
              "connectivity-matrices": {
                "is-allowed": true,
                "path-constraints": {
                  "te-bandwidth": {
                      "generic": "0x1p10",
                  },
                },
                "connectivity-matrix": [
                  {
                    "id": 107,
                    "from": {
                      "tp-ref": "1-0-1",
                    },
                    "to": {
                     "tp-ref": "7-7-0",
                    },
                  },
                  {
                    "id": 308,
                    "from": {
                     "tp-ref": "3-0-3",
                    },
                    "to": {
                     "tp-ref": "8-8-0",
```

```
          },
        },
      ],
    },
  },
  "tunnel-termination-point": [
    {
      "name": "1-0-1",
      "tunnel-tp-id": 10001,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "1-1-0",
      "tunnel-tp-id": 10100,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "2-0-2",
      "tunnel-tp-id": 20002,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "2-2-0",
      "tunnel-tp-id": 20200,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "3-0-3",
      "tunnel-tp-id": 30003,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "3-3-0",
      "tunnel-tp-id": 30300,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "4-0-4",
      "tunnel-tp-id": 40004,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
```

```
      "name": "4-4-0",
      "tunnel-tp-id": 40400,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "5-0-5",
      "tunnel-tp-id": 50005,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "5-5-0",
      "tunnel-tp-id": 50500,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "6-0-6",
      "tunnel-tp-id": 60006,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "6-6-0",
      "tunnel-tp-id": 60600,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "7-0-7",
      "tunnel-tp-id": 70007,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "7-7-0",
      "tunnel-tp-id": 70700,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "8-0-8",
      "tunnel-tp-id": 80008,
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk",
    },
    {
      "name": "8-8-0",
```

```
                    "tunnel-tp-id": 80800,
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk",
                }
            ],
        },
    },
    ]
  },
  ]
}
```

8. **Security Considerations**

The configuration, state, and action data defined in this document
are designed to be accessed via a management protocol with a secure
transport layer, such as NETCONF [RFC6241] or RESTCONF [RFC8040].
The lowest NETCONF layer is the secure transport layer, and the
mandatory-to-implement secure transport is Secure Shell (SSH)
[RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-
to-implement secure transport is TLS [RFC8446].

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

The model presented in this document is used in the interface
between the Customer Network Controller (CNC) and Multi-Domain
Service Coordinator (MDSC), which is referred to as CNC-MDSC
Interface (CMI). Therefore, many security risks such as malicious
attack and rogue elements attempting to connect to various ACTN
components. Furthermore, some ACTN components (e.g., MSDC) represent
a single point of failure and threat vector and must also manage
policy conflicts and eavesdropping of communication between
different ACTN components.

A number of configuration data nodes defined in this document are
writable/deletable (i.e., "config true") These data nodes may be
considered sensitive or vulnerable in some network environments.

These are the subtrees and data nodes and their sensitivity/
vulnerability:

  *ap:

    -ap-id

    -max-bandwidth

    -avl-bandwidth

  *vn-ap:

    -vn-ap-id

    -vn

    -abstract-node

    -ltp

```
    *vn

        -vn-id

        -vn-topology-id

        -abstract-node

    *vnm-id

        -src

        -src-vn-ap-id

        -dest

        -dest-vn-ap-id

        -connectivity-matrix-id
```

9. IANA Considerations

   IANA is requested to make the following allocation for the URIs in
   the "ns" subregistry within the "IETF XML Registry" [RFC3688]:

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

   IANA is requested to make the following allocation for the YANG
   module in the "YANG Module Names" registry [RFC6020]:

```
----------------------------------------------------------------------
name:        ietf-vn
namespace:    urn:ietf:params:xml:ns:yang:ietf-vn
prefix:      vn
reference:   RFC XXXX
----------------------------------------------------------------------
```

10. Acknowledgments

   The authors would like to thank Xufeng Liu, Adrian Farrel, and Tom
   Petch for their helpful comments and valuable suggestions.

   Thanks to Andy Bierman for YANGDIR review.

11. References

11.1. Normative References

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

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

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

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

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

[RFC6991]   Schoenwaelder, J., Ed., "Common YANG Data Types", RFC
            6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-
            editor.org/info/rfc6991>.

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

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

[RFC8174]   Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
            2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
            May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8340]   Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams",
            BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018,
            <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341]   Bierman, A. and M. Bjorklund, "Network Configuration
            Access Control Model", STD 91, RFC 8341, DOI 10.17487/

RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K.,
           and R. Wilton, "Network Management Datastore Architecture
           (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018,
           <https://www.rfc-editor.org/info/rfc8342>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N.,
           Ananthakrishnan, H., and X. Liu, "A YANG Data Model for
           Network Topologies", RFC 8345, DOI 10.17487/RFC8345,
           March 2018, <https://www.rfc-editor.org/info/rfc8345>.

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

[RFC8776] Saad, T., Gandhi, R., Liu, X., Beeram, V., and I.
           Bryskin, "Common YANG Data Types for Traffic
           Engineering", RFC 8776, DOI 10.17487/RFC8776, June 2020,
           <https://www.rfc-editor.org/info/rfc8776>.

[RFC8795] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H.,
           and O. Gonzalez de Dios, "YANG Data Model for Traffic
           Engineering (TE) Topologies", RFC 8795, DOI 10.17487/
           RFC8795, August 2020, <https://www.rfc-editor.org/info/rfc8795>.

11.2. Informative References

[I-D.ietf-ccamp-l1csm-yang] Lee, Y., Lee, K., Zheng, H., Dios, O. G.
           D., and D. Ceccarelli, "A YANG Data Model for L1
           Connectivity Service Model (L1CSM)", Work in Progress,
           Internet-Draft, draft-ietf-ccamp-l1csm-yang-15, 8
           September 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-ccamp-l1csm-yang-15>.

[I-D.ietf-teas-actn-pm-telemetry-autonomics]
           Lee, Y., Dhody, D., Karunanithi, S., Vilalta, R., King,
           D., and D. Ceccarelli, "YANG models for VN/TE Performance
           Monitoring Telemetry and Scaling Intent Autonomics", Work
           in Progress, Internet-Draft, draft-ietf-teas-actn-pm-
           telemetry-autonomics-06, 25 August 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-teas-actn-pm-telemetry-autonomics-06>.

[I-D.ietf-teas-te-service-mapping-yang]
           Lee, Y., Dhody, D., Fioccola, G., Wu, Q., Ceccarelli, D.,
           and J. Tantsura, "Traffic Engineering (TE) and Service
           Mapping Yang Model", Work in Progress, Internet-Draft,

draft-ietf-teas-te-service-mapping-yang-08, 28 August
2021, <https://datatracker.ietf.org/doc/html/draft-ietf-
teas-te-service-mapping-yang-08>.

[I-D.ietf-teas-yang-te] Saad, T., Gandhi, R., Liu, X., Beeram, V.
P., Bryskin, I., and O. G. D. Dios, "A YANG Data Model
for Traffic Engineering Tunnels, Label Switched Paths and
Interfaces", Work in Progress, Internet-Draft, draft-
ietf-teas-yang-te-27, 8 July 2021, <https://
datatracker.ietf.org/doc/html/draft-ietf-teas-yang-
te-27>.

[RFC7926] Farrel, A., Ed., Drake, J., Bitar, N., Swallow, G.,
Ceccarelli, D., and X. Zhang, "Problem Statement and
Architecture for Information Exchange between
Interconnected Traffic-Engineered Networks", BCP 206, RFC
7926, DOI 10.17487/RFC7926, July 2016, <https://www.rfc-
editor.org/info/rfc7926>.

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

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models
Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018,
<https://www.rfc-editor.org/info/rfc8309>.

[RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for
Abstraction and Control of TE Networks (ACTN)", RFC 8453,
DOI 10.17487/RFC8453, August 2018, <https://www.rfc-
editor.org/info/rfc8453>.

[RFC8454] Lee, Y., Belotti, S., Dhody, D., Ceccarelli, D., and B.
Yoon, "Information Model for Abstraction and Control of
TE Networks (ACTN)", RFC 8454, DOI 10.17487/RFC8454,
September 2018, <https://www.rfc-editor.org/info/
rfc8454>.

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

Appendix A. Performance Constraints

At the time of creation of VN, it is natural to provide VN level
constraints and optimization criteria. It should be noted that this
YANG model rely on the TE-Topology Model [RFC8795] by using a

reference to an abstract node to achieve this. Further,
connectivity-matrix structure is used to assign the constraints and
optimization criteria include delay, jitter etc. [RFC8776] define
some of the metric-types already and future documents are meant to
augment it.

Note that the VN compute allows inclusion of the constraints and the
optimization criteria directly in the RPC to allow it to be used
independently.

Appendix B. Contributors Addresses

Qin Wu
Huawei Technologies
Email: bill.wu@huawei.com

Peter Park
KT
Email: peter.park@kt.com

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

Xian Zhang
Huawei Technologies
Email: zhang.xian@huawei.com

Sergio Belotti
Nokia
Email: sergio.belotti@nokia.com

Takuya Miyasaka
KDDI
Email: ta-miyasaka@kddi.com

Kenichi Ogaki
KDDI
Email: ke-oogaki@kddi.com

Authors' Addresses

Young Lee (editor)
Samsung Electronics

Email: younglee.tx@gmail.com

Dhruv Dhody (editor)
Huawei Technologies
Divyashree Techno Park, Whitefield

Bangalore 560066
Karnataka
India

Email: dhruv.ietf@gmail.com

Daniele Ceccarelli
Ericsson
Torshamnsgatan,48
Stockholm, Sweden

Email: daniele.ceccarelli@ericsson.com

Igor Bryskin
Individual

Email: i_bryskin@yahoo.com

Bin Yeong Yoon
ETRI

Email: byyun@etri.re.kr