Abstract

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

This document provides a YANG 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 [L2SM] and L1CSM [L1CSM] 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 VNs in terms of VN Members comprising a VN, multi-source and/or multi-destination characteristics of 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 [TE-Topo] which provides TE network topology abstraction and management operation. Once TE-topology Model is used in triggering VN instantiation over the networks, TE-tunnel [TE-tunnel] 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.

1.2. Tree diagram

A simplified graphical representation of the data model is used in Section 5 of 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.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>vn</td>
<td>ietf-vn</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>nw</td>
<td>ietf-network</td>
<td>[RFC8345]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[TE-Tunnel]</td>
</tr>
<tr>
<td>te-topo</td>
<td>ietf-te-topology</td>
<td>[TE-TOPO]</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>VN-Member 1</th>
<th>L1-L4</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN-Member 2</td>
<td>L1-L7</td>
</tr>
<tr>
<td>VN-Member 3</td>
<td>L2-L4</td>
</tr>
<tr>
<td>VN-Member 4</td>
<td>L3-L8</td>
</tr>
</tbody>
</table>
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. In some cases, more than one abstract nodes can be employed to express their VN.

### 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 2 VN is always built on top of a Type 1 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 on top of the Type 1 VN.

```
+---------------------------------------------------+
| S1                      | S2                      |
| O----------------------| O----------------------|
|                        |                        |
| S3 / \             \ S4       \ S5       |
| L1----O-----------------O--------O--------O--------|------L4
| \                  \ \            \            |
```

```
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 Customer End-Point, respectively.

This VN can be modeled as one abstract node representation as follows:
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  |
+--------+                                +--------+
| POST TE-topo | model(with Conn. | POST /nw:networks/nw:network/|
| model on one | Matrix on one Abstract node | nw:node/te-node-id/tet:connectivity-matrices/tet:connectivity-matrix |
| Abstract node |----------------------------------------> HTTP 200
+---------------+                                +---------------+
| POST the VN identifying AP, VNAP and VN-Members and maps to the TE-topo | POST /VN<----------------------------------------> If there is multi-dest’n module, then MDSC selects a src or dest’n and update VN YANG
| HTTP 200 <----------------------------------------|
+---------------+                                +---------------+
| GET the VN YANG status | GET /VN|---> 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.
Where the following topology is the underlay for Abstraction Node 1 (AN1).

If CNC creates the single abstract topology, the following diagram shows the message flow between CNC and MDSC to instantiate this VN using VN and TE-Topology Model.
On the other hand, if MDSC create single 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    |                        | MDSC   |
+--------+                        +--------+                        +--------+                        +--------+
CNC POST VN Identifying AP, VNAp and VN Members  POST /VN --------------------------> MDSC populates a single Abst. node topology by itself
<---------------------------------------- HTTP 200

CNC POST the VN identifying AP, VNAp and VN Members and maps to the TE-topo  POST /VN -------------------------->
<---------------------------------------- HTTP 200
```
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 [ACTN-INFO]. 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
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.

```yang
++--rw vn
    ++--rw vn-list* [vn-id]
        ++--rw vn-id uint32
        ++--rw vn-name? string
        ++--rw vn-topology-id? te-types:te-topology-id
        ++--rw vn-member-list* [vn-member-id]
            ++--rw vn-member-id uint32
            ++--rw src
                ++--rw src? -> /ap/access-point-list/access-point-id
                ++--rw src-vn-ap-id? -> /ap/access-point-list/vn-ap/vn-ap-id
                ++--rw multi-src? boolean {multi-src-dest}?
                ++--rw dest
                    ++--rw dest? -> /ap/access-point-list/access-point-id
                    ++--rw dest-vn-ap-id? -> /ap/access-point-list/vn-ap/vn-ap-id
                    ++--rw multi-dest? boolean {multi-src-dest}?
                        ++--ro oper-status? identityref
                        ++--ro if-selected? boolean {multi-src-dest}?
                        ++--rw admin-status? identityref
                        ++--ro oper-status? identityref
```

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 [TE-MAP].
The VN Yang model can be extended to support telemetry, performance monitoring and network autonomies as described in [ACTN-PM].

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 ap
    |    +--rw access-point-list* [access-point-id]
    |        +--rw access-point-id       uint32
    |        +--rw access-point-name?    string
    |        +--rw max-bandwidth?        te-types:te-bandwidth
    |        +--rw avl-bandwidth?        te-types:te-bandwidth
    |        +--rw vn-ap* [vn-ap-id]
    |        |    +--rw vn-ap-id            uint32
    |        |    +--rw vn?                 -> /vn/vn-list/vn-id
    |        |    +--rw ltp?                te-types:te-tp-id
    +--rw vn
6. VN YANG Code

The YANG code is as follows:

<CODE BEGINS> file "ietf-vn@2019-02-04.yang"

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

  /* Import network */
  import ietf-network {
    prefix "nw";
  }

  /* Import TE generic types */
  import ietf-te-types {
    prefix "te-types";
  }

  /* Import Abstract TE Topology */
  import ietf-te-topology {
    prefix "tet";
  }

  organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS)
    Working Group";
  contact
    "Editor: Young Lee <leeyoung@huawei.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.

revision 2019-02-04 {
    description
    "initial version.";
    reference
    "TBD";
}

/* Features */

feature multi-src-dest {
    description
"Support for selection of one src or destination among multiple.";
}

/*identity path-metric-delay {
    base te-types:path-metric-type;
    description
    "delay path metric";
}
identity path-metric-delay-variation {
    base te-types:path-metric-type;
    description
    "delay-variation path metric";
}
identity path-metric-loss {
    base te-types:path-metric-type;
    description
    "loss path metric";
}*/

identity vn-state-type {
    description
    "Base identity for VN state";
}

identity vn-state-up {
    base vn-state-type;
    description "VN state up";
}

identity vn-state-down {
    base vn-state-type;
}
typedef vn-disjointness {
  type bits {
    bit node {
      position 0;
      description "node disjoint";
    }
    bit link {
      position 1;
    }
  }
}
description "link disjoint";
}
bit srlg {
  position 2;
  description "srlg disjoint";
}

description
"type of the resource disjointness for
  VN level applied across all VN members
  in a VN";
}

grouping vn-ap {
  description
  "VNAP related information";
leaf vn-ap-id {
  type uint32;
  description
    "unique identifier for the referred VNAP";
}
leaf vn {
  type leafref {
    path "/vn/vn-list/vn-id";
  }
  description
    "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";
}
leaf ltp {
  type te-types:te-tp-id;
  description
    "Reference LTP in the TE-topology";
}
}

grouping access-point{
description
"AP related information";
leaf access-point-id {
  type uint32;
  description
  "unique identifier for the referred access point";
}
leaf access-point-name {
  type string;
  description
  "ap name";
}
leaf max-bandwidth {
  type te-types:te-bandwidth;
  description
  "max bandwidth of the AP";
}
leaf avl-bandwidth {
  type te-types:te-bandwidth;
  description
  "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";
}
} //access-point
grouping vn-member {
  description
  "vn-member is described by this container";
  leaf vn-member-id {
    type uint32;
    description
    "vn-member identifier";
  }
  container src
  {
description "the source of VN Member";
leaf src {
    type leafref {
        path "/ap/access-point-list/access-point-id";
    }
    description "reference to source AP";
}
leaf src-vn-ap-id{
    type leafref {
        path "/ap/access-point-list/vn-ap/vn-ap-id";
    }
    description "reference to source VNAP";
}
leaf multi-src {
    if-feature multi-src-dest;
    type boolean;
    description "Is 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 "/ap/access-point-list/access-point-id";
    }
    description "reference to destination AP";
}
leaf dest-vn-ap-id{
    type leafref {
        path "/ap/access-point-list/vn-ap/vn-ap-id";
    }
    description "reference to dest VNAP";
}
leaf multi-dest {
    if-feature multi-src-dest;
    type boolean;
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
  "reference to connetivity-matrix";
}
} // vn-member
/*
grouping policy {
  description
  "policy related to vn-member-id";
  leaf local-reroute {
    type boolean;
    description
    "Policy to state if reroute
can be done locally";
  }
  leaf push-allowed {
    type boolean;
    description
    "Policy to state if changes
can be pushed to the customer";
  }
  leaf incremental-update {
    type boolean;
    description
    "Policy to allow only the
changes to be reported";
  }
} // policy
*/
grouping vn-policy {
  description
  "policy for VN-level diverisity";
  leaf vn-level-diversity {
    type vn-disjointness;
description
"the type of disjointness on the VN level
(i.e., across all VN members)";
}
} /*
grouping metrics-op {
  description
  "metric related information";
  list metric{
    key "metric-type";
    config false;
    description
    "The list of metrics for VN";
    leaf metric-type {
      type identityref {
        base te-types:path-metric-type;
      }
      description
      "The VN metric type."
    }
    leaf value{
      type uint32;
      description
      "The limit value"
    }
  }
} */ /*
grouping metrics {
  description
  "metric related information";
  list metric{
    key "metric-type";
    description
    "The list of metrics for VN";
    uses te:path-metrics-bounds_config;
    container optimize{
      description
      "optimizing constraints";
      leaf enabled{
        type boolean;
        description
        "Metric to optimize";
      }
    }
  }
} */
leaf value{
    type uint32;
    description
        "The computed value";
}
}
}
*/
/*
grouping service-metric {
    description
        "service-metric";
    uses te:path-objective-function_config;
    uses metrics;
    uses te-types:common-constraints_config;
    uses te:protection-restoration-params_config;
    uses policy;
} // service-metric
*/
/*
* Configuration data nodes
*/

container ap {
    description
        "AP configurations";
    list access-point-list {
        key "access-point-id";
        description
            "access-point identifier";
        uses access-point {
            description
                "access-point information";
        }
    }
}

container vn {
    description
        "VN configurations";
    list vn-list {
        key "vn-id";
        description
            "VN identifier";
    }
}

Lee & Dhody  Expires August 2019  [Page 23]
"a virtual network is identified by a vn-id"
leaf vn-id {
  type uint32;
  description
    "a unique vn identifier";
}
leaf vn-name {
  type string;
  description "vn name";
}
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-list{
  key "vn-member-id";
  description
    "List of VN-members in a VN";
  uses vn-member;
  /*uses metrics-op;*/
  leaf oper-status {
    type identityref {
      base vn-state-type;
    }
    config false;
    description
      "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";
}

/*
container multi-src-dest{
  if-feature multi-src-dest;
  config false;
  description
    "The selected VN Member when multi-src and/or mult-destination is enabled.";
  leaf selected-vn-member{
    type leafref {
      path "/vn/vn-list/vn-member-list
      + "/vn-member-id";
    }
    description
      "The selected VN Member along the set of source and destination configured with multi-source and/or multi-destination";
  }
}
/*
/*uses service-metric;*/
leaf admin-status {
  type identityref {
    base vn-admin-state-type;
  }
  default vn-admin-state-up;
  description "VN administrative state.";
}
leaf oper-status {
  type identityref {
    base vn-state-type;
  }
  config false;
  description "VN operational state.";
}
uses vn-policy;
} // vn-list
} // vn-list
rpc vn-compute{
  description
  "The VN computation without actual instantiation";
  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";
    }
    list vn-member-list{
      key "vn-member-id";
      description
      "List of VN-members in a VN";
      uses vn-member;
    }
    uses vn-policy;
    /*uses service-metric;*/
  }
  output {
    list vn-member-list{
      key "vn-member-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";
      }
      /*uses metrics-op;*/
      leaf compute-status {
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).
o 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;

o 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

{  
  "ap":{
    "access-point-list": [  
      {  
        "access-point-id": 101,  
        "access-point-name": "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"  
          }  
        ]  
      },  
      {  
        "access-point-id": 202,  
        "access-point-name": "202",  
        "vn-ap": [  
          {  
            "vn-ap-id": 20201,  
            "vn": 4,  
            "abstract-node": "D4",  
            "ltp": "1-0-1"  
          }  
        ]  
      }  
    ]  
  }  
}
"vn-ap-id": 20201,
"vn": 1,
"abstract-node": "D1",
"ltp": "2-0-2"
}
},
"access-point-id": 303,
"access-point-name": "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"
  }
}
},
"access-point-id": 440,
"access-point-name": "440",
"vn-ap": [
  {
    "vn-ap-id": 44001,
    "vn": 1,
    "abstract-node": "D1",
    "ltp": "4-4-0"
  }
]
},
"access-point-id": 550,
"access-point-name": "550",
"vn-ap": [
  {
    "vn-ap-id": 55002,
    "vn": 2,
    "abstract-node": "D2",
    "ltp": "5-5-0"
  }
]
{
    "access-point-id": 770,
    "access-point-name": "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"
        }
    ],
}
},
{
    "access-point-id": 880,
    "access-point-name": "880",
    "vn-ap": [
        {
            "vn-ap-id": 88001,
            "vn": 1,
            "abstract-node": "D1",
            "ltp": "8-8-0"
        },
        {
            "vn-ap-id": 88003,
            "vn": 3,
            "abstract-node": "D3",
            "ltp": "8-8-0"
        }
    ],
}

"vn":{
    "vn-list": [
        {
            "vn-id": 1,
            "vn-name": "vn1",
            "vn-topology-id": "te-topology:abstract1",
            "abstract-node": "D1",
            "vn-member-list": [
                {
                    "vn-member-id": 104,
                    "src": {
                        "vn-ap-id": 77001,
                        "vn": 1,
                        "abstract-node": "D1",
                        "ltp": "7-7-0"
                    },
                    ...}]}]}}
"src-vn-ap-id": 10101,
},
"dest": {
  "dest": 880,
  "dest-vn-ap-id": 88001,
},
"connectivity-matrix-id": 108
}
}

"vn-id": 2,
"vn-name": "vn2",
"vn-topology-id": "te-topology:abstract2",
"abstract-node": "D2",
"vn-member-list": [
  {
    "vn-member-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-name": "vn3",
"vn-topology-id": "te-topology:abstract3",
"abstract-node": "D3",
"vn-member-list": [
  {
    "vn-member-id": 104,
    "src": {
      "src": 101,
    },
    "dest": {
      "dest": 440,
      "multi-dest": true
    }
  },
  {
    "vn-member-id": 107,
    "src": {
7.2. TE-topology JSON

```json
{
    "networks": {
        "network": [  
```

Lee & Dhody  Expires August 2019  [Page 33]
{  
    "network-types": {  
        "te-topology": {}  
    },  
    "network-id": "abstract1",  
    "provider-id": 201,  
    "client-id": 600,  
    "te-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": {  
                            "bandwidth-generic": {  
                                "te-bandwidth": {  
                                    "generic": [  
                                        {  
                                            "generic": "0x1p10",  
                                        }  
                                    ]  
                                }  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    ],  
    "disjointness": "node link srlg",  
},  
"connectivity-matrix": [  
    {  
        "id": 104,  
        "from": "1-0-1",  
        "to": "4-4-0"  
    },  
    {  
        "id": 107,  
        "from": "1-0-1",  
        "to": "7-7-0"  
    },  
    {  
        "id": 204,  
        "from": "2-0-2",  
        "to": "4-4-0"  
    },  
    {  
        "id": 308,  
        "from": "3-0-3",  
        "to": "4-4-0"  
    }  
}
"from": "3-0-3",
"to": "8-8-0"
},
{
"id": 108,
"from": "1-0-1",
"to": "8-8-0"
},
]
},
"termination-point": [
{
"tp-id": "1-0-1",
"te-tp-id": 10001,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
},
{
"tp-id": "1-1-0",
"te-tp-id": 10100,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
},
{
"tp-id": "2-0-2",
"te-tp-id": 20002,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
},
]
{
  "tp-id": "2-2-0",
  "te-tp-id": 20200,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "3-0-3",
  "te-tp-id": 30003,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "3-3-0",
  "te-tp-id": 30300,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "4-0-4",
  "te-tp-id": 40004,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
"tp-id": "4-4-0",
"te-tp-id": 40400,
"te": {
   "interface-switching-capability": [
      { "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
}
},
{"tp-id": "5-0-5",
"te-tp-id": 50005,
"te": {
   "interface-switching-capability": [
      { "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
}
},
{"tp-id": "5-5-0",
"te-tp-id": 50500,
"te": {
   "interface-switching-capability": [
      { "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
}
},
{"tp-id": "6-0-6",
"te-tp-id": 60006,
"te": {
   "interface-switching-capability": [
      { "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
}
},
{"tp-id": "6-6-0",\n
"te-tp-id": 60600,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},

"tp-id": "7-0-7",
"te-tp-id": 70007,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},

"tp-id": "7-7-0",
"te-tp-id": 70700,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},

"tp-id": "8-0-8",
"te-tp-id": 80008,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},

"tp-id": "8-8-0",
"te-tp-id": 80800,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},

"network-types": {
    "te-topology": {}
},

"network-id": "abstract2",
"provider-id": 201,
"client-id": 600,
"te-topology-id": "te-topology:abstract2",
"node": [
    {
        "node-id": "D2",
        "te-node-id": "2.0.1.2",
        "te": {
            "te-node-attributes": {
                "domain-id": 1,
                "is-abstract": [null],
                "connectivity-matrices": {
                    "is-allowed": true,
                    "underlay": {
                        "enabled": true
                    },
                    "path-constraints": {
                        "bandwidth-generic": {
                            "te-bandwidth": {
                                "generic": ["0x1p10"
                            ]
                        }
                    },
                    "optimizations": {
                        "objective-function": {
                            "objective-function-type": "of-maximize-residual-bandwidth"
                        }
                    }
                }
            }
        }
    }
]
{
  "id": 105,
  "from": "1-0-1",
  "to": "5-5-0",
  "underlay": {
    "enabled": true,
    "primary-path": {
      "network-ref": "absolute",
      "path-element": [
        {
          "path-element-id": 1,
          "index": 1,
          "numbered-hop": {
            "address": "4.4.4.4",
            "hop-type": "STRICT"
          }
        },
        {
          "path-element-id": 2,
          "index": 2,
          "numbered-hop": {
            "address": "7.7.7.7",
            "hop-type": "STRICT"
          }
        }
      ]
    }
  },
  "termination-point": {
    "tp-id": "1-0-1",
    "te-tp-id": 10001,
    "te": {
      "interface-switching-capability": {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    }
  }
}

}
{
    "tp-id": "1-1-0",
    "te-tp-id": 10100,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "2-0-2",
    "te-tp-id": 20002,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "2-2-0",
    "te-tp-id": 20200,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "3-0-3",
    "te-tp-id": 30003,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
}
"tp-id": "3-3-0",
"te-tp-id": 30300,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},
},
{
"tp-id": "4-0-4",
"te-tp-id": 40004,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},
},
{
"tp-id": "4-4-0",
"te-tp-id": 40400,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},
},
{
"tp-id": "5-0-5",
"te-tp-id": 50005,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},
},
{
"tp-id": "5-5-0",
"te-tp-id": 50005,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
},
}
"te-tp-id": 50500,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{ "tp-id": "6-0-6",
  "te-tp-id": 60006,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},

{ "tp-id": "6-6-0",
  "te-tp-id": 60600,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},

{ "tp-id": "7-0-7",
  "te-tp-id": 70007,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},

{ "tp-id": "7-7-0",
  "te-tp-id": 70700,
"te": {
    "interface-switching-capability": [
        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ],
    "tp-id": "8-0-8",
    "te-tp-id": 80008,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    },
    "network-types": {
        "te-topology": {}
    },
    "network-id": "abstract3",
    "provider-id": 201,
    "client-id": 600,
    "te-topology-id": "te-topology:abstract3",
    "node": [
        {
            "node-id": "D3",
            "te-node-id": "3.0.1.1",
            "network-types": {
                "te-topology": {}
            }
        },
"te": {
  "te-node-attributes": {
    "domain-id": 3,
    "is-abstract": [null],
    "connectivity-matrices": {
      "is-allowed": true,
      "path-constraints": {
        "bandwidth-generic": {
          "te-bandwidth": {
            "generic": [
              {
                "generic": "0x1p10",
              }
            ]
          }
        }
      }
    }
  },
  "connectivity-matrix": {
    "id": 107,
    "from": "1-0-1",
    "to": "7-7-0"
  },
  "id": 308,
  "from": "3-0-3",
  "to": "8-8-0"
},
"termination-point": {
  "tp-id": "1-0-1",
  "te-tp-id": 10001,
  "te": {
    "interface-switching-capability": {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  }
},
  "tp-id": "1-1-0",
  "te-tp-id": 10100,
  "te": {  

"interface-switching-capability": [
    {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
],
,"tp-id": "2-0-2",
"te-tp-id": 20002,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ],
},
,"tp-id": "2-2-0",
"te-tp-id": 20200,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ],
},
,"tp-id": "3-0-3",
"te-tp-id": 30003,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ],
},
,"tp-id": "3-3-0",
"te-tp-id": 30300,
"te": {
  "interface-switching-capability": [
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
}
,
{
"tp-id": "4-0-4",
"te-tp-id": 40004,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
,
{
"tp-id": "4-4-0",
"te-tp-id": 40400,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
,
{
"tp-id": "5-0-5",
"te-tp-id": 50005,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}
,
{
"tp-id": "5-5-0",
"te-tp-id": 50500,
"te": {
"interface-switching-capability": [
{
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]


"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
}
}
{ "tp-id": "6-0-6",
 "te-tp-id": 60006,
 "te": {
   "interface-switching-capability": [
     {
       "switching-capability": "switching-otn",
       "encoding": "lsp-encoding-oduk"
     }
   ]
  }
}]
}
{
 "tp-id": "6-6-0",
 "te-tp-id": 60600,
 "te": {
   "interface-switching-capability": [
     {
       "switching-capability": "switching-otn",
       "encoding": "lsp-encoding-oduk"
     }
   ]
  }
}]
}
{
 "tp-id": "7-0-7",
 "te-tp-id": 70007,
 "te": {
   "interface-switching-capability": [
     {
       "switching-capability": "switching-otn",
       "encoding": "lsp-encoding-oduk"
     }
   ]
  }
}]
}
{
 "tp-id": "7-7-0",
 "te-tp-id": 70700,
 "te": {
   "interface-switching-capability": [
     {
       "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).
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:

- access-point-list:
  - access-point-id
  - max-bandwidth
  - avl-bandwidth

- vn-ap:
  - vn-ap-id
  - vn
  - abstract-node
  - ltp

- vn-list
  - vn-id
  - vn-topology-id
  - abstract-node

- vn-member-id
9. IANA Considerations

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

```
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
```

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

```
name: ietf-vn
reference: RFC XXXX (TDB)
```

10. Acknowledgments

The authors would like to thank Xufeng Liu and Adrian Farrel for their helpful comments and valuable suggestions.
11. References

11.1. Normative References


11.2. Informative References


12. Contributors

Contributor’s Addresses

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

Authors’ Addresses

Young Lee (ed.)
Huawei Technologies
Email: leeyoung@huawei.com

Dhruv Dhody (ed.)
Huawei Technologies
Email: dhruv.ietf@gmail.com
Daniele Ceccarelli
Ericsson
Torshamnsgatan, 48
Stockholm, Sweden
Email: daniele.ceccarelli@ericsson.com

Igor Bryskin
Huawei
Email: Igor.Bryskin@huawei.com

Bin Yeong Yoon
ETRI
Email: byyun@etri.re.kr

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

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