Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 30, 2018

I. Bryskin
Huawei Technologies
X. Liu
Volta Networks
Y. Lee
Huawei Technologies

June 28, 2018

SF Aware TE Topology YANG Model
draft-ietf-teas-sf-aware-topo-model-01

Abstract

This document describes a YANG data model for TE network topologies that are network service and function aware.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 30, 2018.

Copyright Notice

Table of Contents

1. Introduction

Today a network offers to its clients far more services than just connectivity across the network. Large variety of physical, logical and/or virtual service functions, network functions and transport functions (collectively named in this document as SFs) could be allocated for and assigned to a client. As described in [I-D.ietf-teas-use-cases-sf-aware-topo-model], there are some important use cases, in which the network needs to represent to the client SFs at the client's disposal as topological elements in relation to other elements of a topology (i.e. nodes, links, link and tunnel termination points) used by the network to describe itself to the client. Not only would such information allow for the client to auto-discover the network's SFs available for the services provisioned for the client, it would also allow for the client selecting the SFs, duel-optimizing the selection on the SF location on the network and connectivity means (e.g. TE tunnels) to inter-connect the SFs. Consequently thus would give to both the network and the client powerful means for the service function chain (SFC [RFC7498] [RFC7665]) negotiation to achieve most efficient and cost effective (from the network point of view) and most optimal yet satisfying all necessary constraints of SFCs(from the client's point of view).

This document defines a YANG data model that allows service functions to be represented along with TE topology elements.

1.1. Terminology

The keywords "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].

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

- augment
- data model
- data node

1.2. Tree Diagrams

A simplified graphical representation of the data model is presented in this document, by using the tree format defined in [I-D.ietf-netmod-yang-tree-diagrams].

1.3. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
inet	ietf-inet-types	[RFC6991]
nw	ietf-network	[I-D.ietf-i2rs-yang-network-topo]
nt	ietf-network-topology	[I-D.ietf-i2rs-yang-network-topo]
tet	ietf-te-topology	[I-D.ietf-teas-yang-te-topo]

Table 1: Prefixes and Corresponding YANG Modules

2. Modeling Considerations

The model introduced in this document is an augmentation of the TE Topology model defined in [I-D.ietf-teas-yang-te-topo]. SFs are modeled as child elements of a TE node similarly to how Link Termination Points (LTPs) and Tunnel Termination Points (TTPs) are modeled in the TE Topology model. The SFs are defined as opaque objects identified via topology unique service-function-id's. Each SF has one or more Connection Points (CPs) identified via SF-unique sf-connection-point-id's, over which the SF could be connected to other SFs resided on the same TE node, as well as to other elements of the TE node, in particular, to the node's LTPs and/or TTPs. An interested client may use service-function-id's to look up the SFs in TOSCA or YANG data store(s) defined by [ETSI-NFV-MAN] to retrieve the details of the SFs, for example, to understand the SF's mutual substitutability.

The TE Topology model introduces a concept of Connectivity Matrix (CM), and uses the CM to describe which and at what costs a TE node's LTPs could be inter-connected internally across the TE node. The model defined in this document heavily uses the same concept to describe the SF connectivity via introducing 3 additional CMs:

1. SF2SF CM. This CM describes which pairs of SFs could be locally inter-connected, and, if yes, in which direction, via which CPs and at what costs. In other words, the SF2SF CM describes how SFs residing on the same TE node could be inter-connected into local from the TE node's perspective SFCs;

2. SF2LTP CM. This CM describes how, in which direction and at what costs the TE node's SFs could be connected to the TE node's LTPs and hence to SFs residing on neighboring TE nodes that are connected to LTPs at the remote ends of corresponding TE links;

3. SF2TTP CM. This CM describes how, in which direction and at what costs the TE node's SFs could be connected to the TE node's TTPs and hence to SFs residing on other TE nodes on the topology that could be inter-connected with the TE node in question via TE tunnels terminated by the corresponding TTPs.

In addition to SF2SF CM, the local SF chaining could be described with the help of ETSI models Virtual Links (VLs) [ETSI-NFV-MAN]. This option is especially useful when the costs of the local chaining are negligible as compared to ones of the end-to-end SFCs said local SFCs are part of.

Section 3 and 4 provide the YANG model structure and the YANG module for SF-aware Topology. Section 5 and 6 provide the YANG model

structure and the YANG module for Data Center Compute Node resource abstraction. This provides an example of SF2LTP CM where DC compute nodes are connected to LTPs at the remote ends of the corresponding TE links. This use-case is described in Section 12 of [I-D.ietf-teas-use-cases-sf-aware-topo-model].

3. Model Structure

```
module: ietf-te-topology-sf
  augment /nw:networks/nw:network/nw:network-types/tet:te-topology:
    +--rw sf!
  augment /nw:networks/nw:network/nw:node/tet:te
/tet:te-node-attributes:
    +--rw service-function
       +--rw connectivity-matrices
       |  +--rw connectivity-matrix* [id]
       |     +--rw id                 uint32
       |     +--rw from
       |     |  +--rw service-function-id?      string
       |     |  +--rw sf-connection-point-id?   string
       |     +--rw to
       |     |  +--rw service-function-id?      string
       |     |  +--rw sf-connection-point-id?   string
       |     +--rw enabled?           boolean
       |     +--rw direction?         connectivity-direction
       |     +--rw virtual-link-id?   string
       +--rw link-terminations
          +--rw link-termination* [id]
             +--rw id           uint32
             +--rw from
             |  +--rw tp-ref?   -> ../../../../../../../..
/nt:termination-point/tp-id
             +--rw to
             |  +--rw service-function-id?      string
             |  +--rw sf-connection-point-id?   string
             +--rw enabled?     boolean
             +--rw direction?   connectivity-direction
  augment /nw:networks/nw:network/nw:node/tet:te
/tet:information-source-entry:
    +--ro service-function
       +--ro connectivity-matrices
       |  +--ro connectivity-matrix* [id]
       |     +--ro id                 uint32
       |     +--ro from
       |     |  +--ro service-function-id?      string
       |     |  +--ro sf-connection-point-id?   string
       |     +--ro to
       |     |  +--ro service-function-id?      string
```

```
          |     |  +--ro sf-connection-point-id?   string
          |     +--ro enabled?           boolean
          |     +--ro direction?         connectivity-direction
          |     +--ro virtual-link-id?   string
          +--ro link-terminations
             +--ro link-termination* [id]
                +--ro id           uint32
                +--ro from
                +--ro to
                |  +--ro service-function-id?      string
                |  +--ro sf-connection-point-id?   string
                +--ro enabled?     boolean
                +--ro direction?   connectivity-direction
    augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point:
       +--rw service-function
          +--rw tunnel-terminations
             +--rw tunnel-termination* [id]
                +--rw id                          uint32
                +--rw service-function-id?        string
                +--rw sf-connection-point-id?     string
                +--rw enabled?                    boolean
                +--rw direction?                  connectivity-direction
```

4. YANG Modules

```
   <CODE BEGINS> file "ietf-te-topology-sf@2018-02-27.yang"
   module ietf-te-topology-sf {
     yang-version 1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-sf";

     prefix "tet-sf";

     import ietf-network {
       prefix "nw";
     }

     import ietf-network-topology {
       prefix "nt";
     }

     import ietf-te-topology {
       prefix "tet";
     }

     organization
```

```
       "Traffic Engineering Architecture and Signaling (TEAS)
        Working Group";

     contact
       "WG Web:   <http://tools.ietf.org/wg/teas/>
        WG List:  <mailto:teas@ietf.org>

        Editors:  Igor Bryskin
                  <mailto:Igor.Bryskin@huawei.com>

                  Xufeng Liu
                  <mailto:Xufeng_Liu@jabil.com>";

     description
       "Network service and function aware aware TE topology model.

        Copyright (c) 2018 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
        (http://trustee.ietf.org/license-info).";

     revision 2018-02-27 {
       description "Initial revision";
       reference "TBD";
     }

     /*
      * Typedefs
      */
     typedef connectivity-direction {
       type enumeration {
         enum "to" {
           description
           "The direction is uni-directional, towards the 'to'
            entity direction.";
         }
         enum "from" {
           description
           "The direction is uni-directional, from the 'to'
            entity direction.";
         }
         enum "bidir" {
           description
```

```
          "The direction is bi-directional.";
        }
      }
      description
        "A type used to indicates whether a connectivity is
         uni-directional, or bi-directional. If the relation is
         uni-directional, the value of this type indicates the
         direction.";
    } // connectivity-direction

    /*
     * Groupings
     */
    grouping service-function-node-augmentation {
      description
        "Augmenting a TE node to be network service and function
         aware.";
      container service-function {
        description
          "Containing attributes related to network services and
           network functions";
        container connectivity-matrices {
          description
            "Connectivity relations between network services/functions
             on a TE node, which can be either abstract or physical.";
          reference
            "ETSI GS NFV-MAN 01: Network Functions Virtualisation
             (NFV); Management and Orchestration.
             RFC7665: Service Function Chaining (SFC) Architecture.";
          list connectivity-matrix {
            key "id";
            description
              "Represents the connectivity relations between network
               services/functions on a TE node.";
            leaf id {
              type uint32;
              description "Identifies the connectivity-matrix entry.";
            }

            container from {
              description
                "Reference to the source network service or
                 network function.";
              leaf service-function-id {
                type string;
                description
                  "Reference to a network service or a network
                   function.";
```

```
            }
            leaf sf-connection-point-id {
              type string;
              description
                "Reference to a connection point on a network
                 service or a network function.";
            }
          } // from
          container to {
            description
              "Reference to the destination network service or
               network function.";
            leaf service-function-id {
              type string;
              description
                "Reference to a network service or a network
                 function.";
              }
            leaf sf-connection-point-id {
              type string;
              description
                "Reference to a connection point on a network
                 service or a network function.";
            }
          } // to
          leaf enabled {
            type boolean;
            description
              "'true' if this connectivity entry is enabled.";
          }
          leaf direction {
            type connectivity-direction;
            description
              "Indicates whether this connectivity is
               uni-directional, or bi-directional. If the
               relation is uni-directional, the value of
               this leaf indicates the direction.";
          }
          leaf virtual-link-id {
            type string;
            description
              "Reference to a virtual link that models this
               conectivity relation in the network function
               model.";
          }
        } // connectivity-matrix
      } // connectivity-matrices
```

```
        container link-terminations {
          description
            "Connectivity relations between network services/functions
             and link termination points on a TE node, which can be
             either abstract or physical.";
          reference
            "ETSI GS NFV-MAN 01: Network Functions Virtualisation
             (NFV); Management and Orchestration.
             RFC7665: Service Function Chaining (SFC) Architecture.";
          list link-termination {
            key "id";
            description
              "Each entry of the list represents the connectivity
               relation between a network service/function and
               a link termination point on a TE node.";
            leaf id {
              type uint32;
              description "Identifies the termination entry.";
            }

            container from {
              description
                "Reference to the link termination point.";
            } // from
            container to {
              description
                "Reference to the network service or network
                 function.";
              leaf service-function-id {
                type string;
                description
                  "Reference to a network service or a network
                   function.";
                }
              leaf sf-connection-point-id {
                type string;
                description
                  "Reference to a connection point on a network
                   service or a network function.";
              }
            } // to
            leaf enabled {
              type boolean;
              description
                "'true' if this connectivity entry is enabled.";
            }
            leaf direction {
              type connectivity-direction;
```

```
              description
                "Indicates whether this connectivity is
                 uni-directional, or bi-directional. If the
                 relation is uni-directional, the value of
                 this leaf indicates the direction.";
            }
          } // link-termination
        }
      }
    } // service-function-node-augmentation

    grouping service-function-ttp-augmentation {
      description
        "Augmenting a tunnel termination point to be network service
         aware.";
      container service-function {
        description
          "Containing attributes related to network services and
           network functions";
        container tunnel-terminations {
          description
            "Connectivity relations between network services/functions
             and tunnel termination points on a TE node, which can be
             either abstract or physical.";
          reference
            "ETSI GS NFV-MAN 01: Network Functions Virtualisation
             (NFV); Management and Orchestration.
             RFC7665: Service Function Chaining (SFC) Architecture.";
          list tunnel-termination {
            key "id";
            description
              "Each entry of the list represents the connectivity
               relation between a network service/function and
               a tunnel termination point on a TE node.";
            leaf id {
              type uint32;
              description "Identifies the termination entry.";
            }

            leaf service-function-id {
              type string;
              description
                "Reference to a network service or a network
                 function.";
            }
            leaf sf-connection-point-id {
              type string;
              description
```

```
                "Reference to a connection point on a network
                 service or a network function.";
            }
            leaf enabled {
              type boolean;
              description
                "'true' if this connectivity entry is enabled.";
            }
            leaf direction {
              type connectivity-direction;
              description
                "Indicates whether this connectivity is
                 uni-directional, or bi-directional. If the
                 relation is uni-directional, the value of
                 this leaf indicates the direction.";
            }
          } // link-termination
        }
      }
    } // service-function-ttp-augmentation

    grouping sf-topology-type {
      description
        "Identifies the SF aware TE topology type.";
      container sf {
        presence "Indidates that the TE topology is SF aware.";
        description
          "Its presence identifies that the TE topology is SF aware.";
      }
    } // sf-topology-type

    /*
     * Augmentations
     */
    /* Augmentations to network-types/te-topology */
    augment "/nw:networks/nw:network/nw:network-types/"
      + "tet:te-topology" {
      description
        "Defines the SF aware TE topology type.";
      uses sf-topology-type;
    }

    /* Augmentations to te-node-attributes */
    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:te-node-attributes" {
      description
        "Parameters for SF aware TE topology.";
      uses service-function-node-augmentation;
```

```
    }

    augment "/nw:networks/nw:network/nw:node/tet:te/"
          + "tet:information-source-entry" {
      description
        "Parameters for SF aware TE topology.";
      uses service-function-node-augmentation;
    }

    /* Augmentations to tunnel-termination-point */
    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:tunnel-termination-point" {
      description
        "Parameters for SF aware TE topology.";
      uses service-function-ttp-augmentation;
    }

    /* Augmentations to connectivity-matrix */
    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:te-node-attributes/tet-sf:service-function/"
      + "tet-sf:link-terminations/tet-sf:link-termination/"
      + "tet-sf:from" {
      description
        "Add reference to the link termination point.
         This portion cannot be shared with the state module.";
      leaf tp-ref {
        type leafref {
          path "../../../../../../../nt:termination-point/"
            + "nt:tp-id";
        }
        description
          "Reference to the link termination point.";
      }
    }
  }
  <CODE ENDS>
```

5. Model Structure

```
  module: ietf-cso-dc
      +--rw cso
         +--rw dc* [id]
         |  +--rw hypervisor* [id]
         |  |  +--rw ram
         |  |  |  +--rw total?   uint32
         |  |  |  +--rw used?    uint32
```

```
          |  |  |  +--rw free?    uint32
          |  |  +--rw disk
          |  |  |  +--rw total?   uint32
          |  |  |  +--rw used?    uint32
          |  |  |  +--rw free?    uint32
          |  |  +--rw vcpu
          |  |  |  +--rw total?   uint16
          |  |  |  +--rw used?    uint16
          |  |  |  +--rw free?    uint16
          |  |  +--rw instance*   -> /cso/dc/instance/id
          |  |  +--rw id          string
          |  |  +--rw name?       string
          |  +--rw instance* [id]
          |  |  +--rw flavor
          |  |  |  +--rw disk?    uint32
          |  |  |  +--rw ram?     uint32
          |  |  |  +--rw vcpus?   uint16
          |  |  |  +--rw id?      string
          |  |  |  +--rw name?    string
          |  |  +--rw image
          |  |  |  +--rw checksum    string
          |  |  |  +--rw size        uint32
          |  |  |  +--rw format
          |  |  |  |  +--rw container?   enumeration
          |  |  |  |  +--rw disk?        enumeration
          |  |  |  +--rw id?         string
          |  |  |  +--rw name?       string
          |  |  +--rw hypervisor?   -> /cso/dc/hypervisor/id
          |  |  +--rw port*         -> /cso/dc/network/subnetwork/port
   /id
          |  |  +--rw project?      string
          |  |  +--rw status?       enumeration
          |  |  +--rw id            string
          |  |  +--rw name?         string
          |  +--rw image* [id]
          |  |  +--rw checksum    string
          |  |  +--rw size        uint32
          |  |  +--rw format
          |  |  |  +--rw container?   enumeration
          |  |  |  +--rw disk?        enumeration
          |  |  +--rw id          string
          |  |  +--rw name?       string
          |  +--rw flavor* [id]
          |  |  +--rw disk?    uint32
          |  |  +--rw ram?     uint32
          |  |  +--rw vcpus?   uint16
          |  |  +--rw id       string
          |  |  +--rw name?    string
```

```
          |  +--rw dc-monitoring-param* [name]
          |  |  +--rw name            string
          |  |  +--rw value-string?   string
          |  +--rw network* [id]
          |  |  +--rw subnetwork* [id]
          |  |  |  +--rw port* [id]
          |  |  |  |  +--rw ip-address?   inet:ip-address
          |  |  |  |  +--rw instance?     -> /cso/dc/instance/id
          |  |  |  |  +--rw project?      string
          |  |  |  |  +--rw status?       enumeration
          |  |  |  |  +--rw id            string
          |  |  |  |  +--rw name?         string
          |  |  |  +--rw project?   string
          |  |  |  +--rw status?    enumeration
          |  |  |  +--rw id         string
          |  |  |  +--rw name?      string
          |  |  +--rw dhcp-agent* [id]
          |  |  |  +--rw enabled?   boolean
          |  |  |  +--rw pools* [ip-address]
          |  |  |  |  +--rw ip-address    inet:ip-address
          |  |  |  +--rw project?   string
          |  |  |  +--rw status?    enumeration
          |  |  |  +--rw id         string
          |  |  |  +--rw name?      string
          |  |  +--rw project?      string
          |  |  +--rw status?       enumeration
          |  |  +--rw id            string
          |  |  +--rw name?         string
          |  |  +--rw cso-ref?      -> /cso/cso-id
          |  +--rw ap*                    -> /actn-vn:actn/ap
   /access-point-list/access-point-id
          |  +--rw cso-ref?               -> /cso/cso-id
          |  +--rw id                     string
          |  +--rw name?                  string
          +--rw cso-id?   string
```

6. YANG Modules

```
<CODE BEGINS> file "ietf-cso-dc@2017-01-16.yang"
module ietf-cso-dc
{
  namespace "urn:ietf:params:xml:ns:yang:ietf-cso-dc";
  prefix "dc";

  import ietf-inet-types {
    prefix "inet";
```

```
}

import ietf-actn-vn {
  prefix "actn-vn";
}

revision 2017-01-16 {
  description
    "Initial revision. This YANG file defines
     the reusable base types for CSO DC description.";
  reference
    "Derived from earlier versions of base YANG files";
}

// Abstract models
grouping resource-element {
  leaf id { type string; }
  leaf name { type string; }
}

grouping resource-instance {
  leaf project{ type string; }
  leaf status {
    type enumeration {
      enum active;
      enum inactive;
      enum pending;
    }
  }
  uses resource-element;
}

// Compute models
grouping format {
  leaf container {
    type enumeration {
      enum ami;
      enum ari;
      enum aki;
      enum bare;
      enum ovf;
    }
    default bare;
  }
  leaf disk {
    type enumeration {
      enum ami;
      enum ari;
```

```
            enum aki;
            enum vhd;
            enum vmdk;
            enum raw;
            enum qcow2;
            enum vdi;
            enum iso;
          }
          default qcow2;
        }
      }

      grouping image {
        leaf checksum { type string; mandatory true; }
        leaf size { type uint32; units 'Bytes'; mandatory true; }

        container format {
          uses format;
        }

        uses resource-element;
      }

      grouping flavor {
        leaf disk  { type uint32; units 'GB'; default 0; }
        leaf ram   { type uint32; units 'MB'; default 0; }
        leaf vcpus { type uint16; default 0; }
        uses resource-element;
      }

      grouping ram {
        leaf total { type uint32; units 'MB'; }
        leaf used { type uint32; units 'MB'; }
        leaf free { type uint32; units 'MB'; }
      }

      grouping disk {
        leaf total { type uint32; units 'GB'; }
        leaf used { type uint32; units 'GB'; }
        leaf free { type uint32; units 'GB'; }
      }

      grouping vcpu {
        leaf total { type uint16; }
        leaf used { type uint16; }
        leaf free { type uint16; }
      }
```

```
     grouping hypervisor {

       container ram {
         uses ram;
       }

       container disk {
         uses disk;
       }

       container vcpu {
         uses vcpu;
       }

       leaf-list instance {
         type leafref { path '/cso/dc/instance/id'; } }
       uses resource-element;
     }

     grouping instance {
       container flavor { uses flavor; }
       container image { uses image; }
       leaf hypervisor {
         type leafref { path '/cso/dc/hypervisor/id'; } }
       leaf-list port { type leafref {
           path '/cso/dc/network/subnetwork/port/id'; } }
       uses resource-instance;
     }

     grouping dc-monitoring-param {
       leaf name {
         description "dc-monitoring-param identifier"; type string; }
       leaf value-string {
         description
           "Current value for a string parameter";
         type string;
       }
     }

     grouping dc {

       list hypervisor {
         key id;
         uses hypervisor;
       }

       list instance {
         key id;
```

```
        uses instance;
      }

      list image {
        key id;
        uses image;
      }

      list flavor {
        key id;
        uses flavor;
      }

      list dc-monitoring-param {
          key "name";
          uses dc-monitoring-param;
      }

      list network {
          key id;
          uses network;
      }

      leaf-list ap { type leafref {
          path
            '/actn-vn:actn/actn-vn:ap/actn-vn:access-point-list/'
            + 'actn-vn:access-point-id';
        }
      }
      leaf cso-ref { type leafref { path "/cso/cso-id"; } }
      uses resource-element;
    }



    container cso {
      list dc {
          key id;
          uses dc;
      }

      leaf cso-id { type string; }
    }

    // Network models
    grouping ip-address {
      leaf ip-address { type inet:ip-address; }
    }
```

```
     grouping dhcp-agent {
       leaf enabled { type boolean; }
       list pools {
         key ip-address;
         uses ip-address;
       }
       uses resource-instance;
     }

     grouping network {
       list subnetwork {
         key id;
         uses subnetwork;
       }
       list dhcp-agent {
         key id;
         uses dhcp-agent;
       }
       uses resource-instance;
       leaf cso-ref { type leafref { path "/cso/cso-id"; } }
     }

     grouping subnetwork {
       list port {
         key id;
         uses port;
       }
       uses resource-instance;
     }

     grouping port {
       leaf ip-address { type inet:ip-address; }
       leaf instance { type leafref { path '/cso/dc/instance/id'; } }
       uses resource-instance;
     }



   }
   <CODE ENDS>
```

7. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

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

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

---------------------------------------------------------------------
URI: urn:ietf:params:xml:ns:yang:ietf-te-topology-sf-state
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 [RFC7950]:

```
---------------------------------------------------------------------
name:         ietf-te-topology-sf
namespace:    urn:ietf:params:xml:ns:yang:ietf-te-topology-packet
prefix:       tet-sf
reference:    RFC XXXX
---------------------------------------------------------------------

---------------------------------------------------------------------
name:         ietf-te-topology-sf-state
namespace:  urn:ietf:params:xml:ns:yang:ietf-te-topology-packet-state
prefix:       tet-sf-s
reference:    RFC XXXX
---------------------------------------------------------------------
```

8. Security Considerations

The configuration, state, action and notification data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241]. The data-model by itself does not create any security implications. The security considerations for the NETCONF protocol are applicable. The NETCONF protocol used for sending the data supports authentication and encryption.

9. References

9.1. 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>.

[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>.

[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>.

[ETSI-NFV-MAN] ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration", ETSI GS NFV-MAN 001 V1.1.1, December 2014.

[I-D.ietf-teas-use-cases-sf-aware-topo-model] Bryskin, I., Liu, X., Guichard, J., Lee, Y., Contreras, L., Ceccarelli, D., and J. Tantsura, "Use Cases for SF Aware Topology Models", draft-ietf-teas-use-cases-sf-aware-topo-model-00 (work in progress), June 2018.

[I-D.ietf-i2rs-yang-network-topo] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo-20 (work in progress), December 2017.

[I-D.ietf-teas-yang-te-topo] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-16 (work in progress), June 2018.

[I-D.ietf-netmod-revised-datastores] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-10 (work in progress), January 2018.

9.2. Informative References

[RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <https://www.rfc-editor.org/info/rfc7498>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[I-D.ietf-netmod-yang-tree-diagrams] Bjorklund, M. and L. Berger, "YANG Tree Diagrams", draft-ietf-netmod-yang-tree-diagrams-06 (work in progress), February 2018.

Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module ietf-te-topology-sf defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [I-D.ietf-netmod-revised-datastores]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module, ietf-te-topology-sf-state, is defined as state model, which mirrors the module ietf-te-topology-sf defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion module, ietf-te-topology-sf-state, is redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion module mirrors that of the coorespinding NMDA model, the YANG tree of the companion module is not depicted separately.

A.1. SF Aware TE Topology State Module

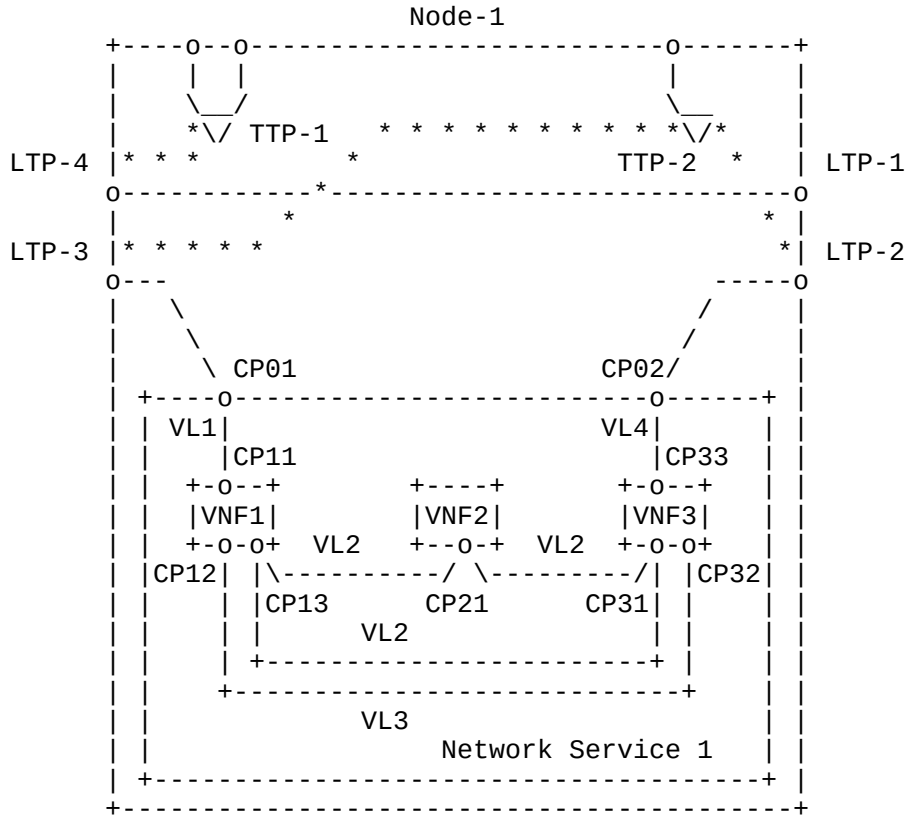
```
<CODE BEGINS> file "ietf-te-topology-sf-state@2018-02-27.yang"
module ietf-te-topology-sf-state {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-sf-state";

  prefix "tet-sf-s";

  import ietf-te-topology-sf {
    prefix "tet-sf";
  }

  import ietf-network-state {
    prefix "nw-s";
  }

  import ietf-network-topology-state {
    prefix "nt-s";
  }

  import ietf-te-topology-state {
    prefix "tet-s";
  }

  organization
    "Traffic Engineering Architecture and Signaling (TEAS)
```

```
        Working Group";

      contact
        "WG Web:   <http://tools.ietf.org/wg/teas/>
         WG List:  <mailto:teas@ietf.org>

         Editors:  Igor Bryskin
                   <mailto:Igor.Bryskin@huawei.com>

                   Xufeng Liu
                   <mailto:Xufeng_Liu@jabil.com>";

      description
        "Network service and function aware aware TE topology operational
         state model for non-NMDA compliant implementations.

         Copyright (c) 2018 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
         (http://trustee.ietf.org/license-info).";

      revision 2018-02-27 {
        description "Initial revision";
        reference "TBD";
      }

      /*
       * Augmentations
       */
      /* Augmentations to network-types/te-topology */
      augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
        + "tet-s:te-topology" {
        description
          "Defines the SF aware TE topology type.";
        uses tet-sf:sf-topology-type;
      }

      /* Augmentations to connectivity-matrix */
      augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
        + "tet-s:te-node-attributes" {
        description
          "Parameters for SF aware TE topology.";
        uses tet-sf:service-function-node-augmentation;
```

```
    }

    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
          + "tet-s:information-source-entry" {
      description
        "Parameters for SF aware TE topology.";
      uses tet-sf:service-function-node-augmentation;
    }

    /* Augmentations to tunnel-termination-point */
    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
      + "tet-s:tunnel-termination-point" {
      description
        "Parameters for SF aware TE topology.";
      uses tet-sf:service-function-ttp-augmentation;
    }

    /* Augmentations to connectivity-matrix */
    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
      + "tet-s:te-node-attributes/tet-sf-s:service-function/"
      + "tet-sf-s:link-terminations/tet-sf-s:link-termination/"
      + "tet-sf-s:from" {
      description
        "Add reference to the link termination point.
         This portion cannot be shared with the state module.";
      leaf tp-ref {
        type leafref {
          path "../../../../../../../nt-s:termination-point/"
            + "nt-s:tp-id";
        }
        description
          "Reference to the link termination point.";
      }
    }
  }
  <CODE ENDS>
```

Appendix B. Data Examples

B.1. A Topology with Multiple Connected Network Functions



The configuration instance data for Node-1 in the above figure could be as follows:

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {
            "sf": {}
          }
        },
        "network-id": "network-sf-aware",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:network-sf-aware",
        "node": [
          {
            "node-id": "Node-1",
```

```
                "te-node-id": "2.0.1.1",
                "te": {
                  "te-node-attributes": {
                    "domain-id": 1,
                    "is-abstract": [null],
                    "connectivity-matrices": {
                    },
                    "service-function": {
                      "connectivity-matrices": {
                        "connectivity-matrix": [
                          {
                            "id": 10,
                            "from": {
                              "service-function-id": "Network Service 1",
                              "sf-connection-point-id": "CP01"
                            },
                            "to": {
                              "service-function-id": "VNF1",
                              "sf-connection-point-id": "CP11"
                            }
                            "direction": "bidir",
                            "virtual-link-id": "VL1"
                          },
                          {
                            "id": 13,
                            "from": {
                              "service-function-id": "VNF1",
                              "sf-connection-point-id": "CP12"
                            },
                            "to": {
                              "service-function-id": "VNF3",
                              "sf-connection-point-id": "CP32"
                            }
                            "direction": "bidir",
                            "virtual-link-id": "VL3"
                          },
                          {
                            "id": 12,
                            "from": {
                              "service-function-id": "VNF1",
                              "sf-connection-point-id": "CP13"
                            },
                            "to": {
                              "service-function-id": "VNF2",
                              "sf-connection-point-id": "CP21"
                            }
                            "direction": "bidir",
                            "virtual-link-id": "VL2"
```

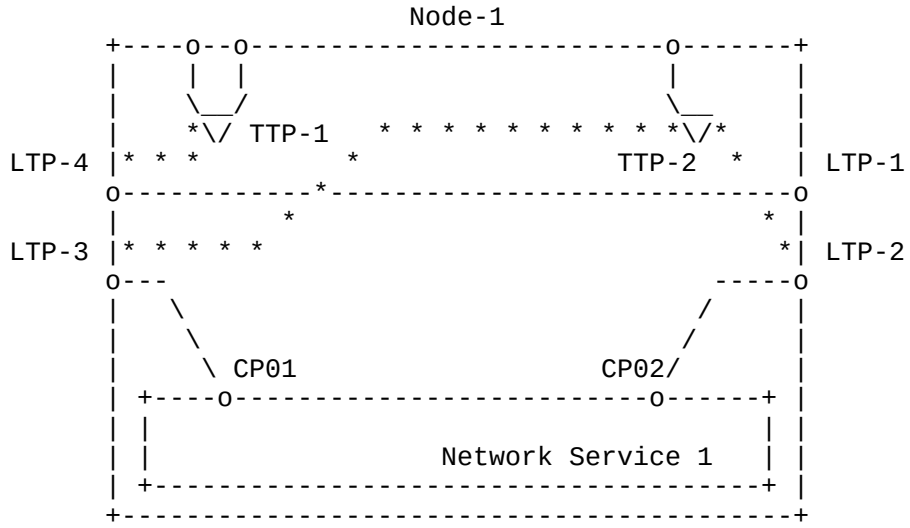
```
        },
        {
          "id": 23,
          "from": {
            "service-function-id": "VNF2",
            "sf-connection-point-id": "CP21"
          },
          "to": {
            "service-function-id": "VNF3"
            "sf-connection-point-id": "CP31"
          }
          "direction": "bidir",
          "virtual-link-id": "VL2"
        },
        {
          "id": 30,
          "from": {
            "service-function-id": "Network Service 1",
            "sf-connection-point-id": "CP02"
          },
          "to": {
            "service-function-id": "VNF3",
            "sf-connection-point-id": "CP33"
          }
          "direction": "bidir",
          "virtual-link-id": "VL4"
        }
      ]
    },
    "link-terminations": {
      "link-termination": [
        {
          "id": 2,
          "from": {
            "tp-ref": "LTP-2"
          },
          "to": {
            "service-function-id": "Network Service 1",
            "sf-connection-point-id": "CP02"
          }
          "direction": "bidir"
        },
        {
          "id": 3,
          "from": {
            "tp-ref": "LTP-3"
          },
          "to": {
```

```
                        "service-function-id": "Network Service 1",
                        "sf-connection-point-id": "CP01"
                      }
                      "direction": "bidir"
                    }
                  ]
                }
              }
            }
            "tunnel-termination-point": [
              {
                "tunnel-tp-id": 10001,
                "name": "TTP-1",
                "service-function-terminations": {
                }
              },
              {
                "tunnel-tp-id": 10002,
                "name": "TTP-2",
                "service-function-terminations": {
                }
              }
            ]
          },
          "termination-point": [
            {
              "tp-id": "LTP-1",
              "te-tp-id": 10001
              "te": {
                "interface-switching-capability": [
                  {
                    "switching-capability": "switching-l2sc",
                    "encoding": "lsp-encoding-ethernet"
                  }
                ]
              }
            },
            {
              "tp-id": "LTP-2",
              "te-tp-id": 10002
              "te": {
                "interface-switching-capability": [
                  {
                    "switching-capability": "switching-l2sc",
                    "encoding": "lsp-encoding-ethernet"
                  }
                ]
              }
```

```
                },
                {
                  "tp-id": "LTP-3",
                  "te-tp-id": 10003
                  "te": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                },
                {
                  "tp-id": "LTP-4",
                  "te-tp-id": 10004
                  "te": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                }
              ]
            }
          ]
        }
      ]
    }
  }
```

B.2. A Topology with an Encapsulated Network Service

In this example, a network service consists of several inter-connected network functions (NFs), and is represented by this model as an encapsulated opaque object without the details between its internals.



The configuration instance data for Node-1 in the above figure could be as follows:

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {
            "sf": {}
          }
        },
        "network-id": "network-sf-aware",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:network-sf-aware",
        "node": [
          {
            "node-id": "Node-1",
            "te-node-id": "2.0.1.1",
            "te": {
              "te-node-attributes": {
                "domain-id": 1,
                "is-abstract": [null],
                "connectivity-matrices": {
                },
                "service-function": {
                  "connectivity-matrices": {
                  },
```

```
                    "link-terminations": {
                      "link-termination": [
                        {
                          "id": 2,
                          "from": {
                            "tp-ref": "LTP-2"
                          },
                          "to": {
                            "service-function-id": "Network Service 1",
                            "sf-connection-point-id": "CP02"
                          }
                          "direction": "bidir"
                        },
                        {
                          "id": 3,
                          "from": {
                            "tp-ref": "LTP-3"
                          },
                          "to": {
                            "service-function-id": "Network Service 1",
                            "sf-connection-point-id": "CP01"
                          }
                          "direction": "bidir"
                        }
                      ]
                    }
                  }
                }
                "tunnel-termination-point": [
                  {
                    "tunnel-tp-id": 10001,
                    "name": "TTP-1",
                    "service-function-terminations": {
                    }
                  },
                  {
                    "tunnel-tp-id": 10002,
                    "name": "TTP-2",
                    "service-function-terminations": {
                    }
                  }
                ]
              },
              "termination-point": [
                {
                  "tp-id": "LTP-1",
                  "te-tp-id": 10001
                  "te": {
```

```
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                },
                {
                  "tp-id": "LTP-2",
                  "te-tp-id": 10002
                  "te": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                },
                {
                  "tp-id": "LTP-3",
                  "te-tp-id": 10003
                  "te": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                },
                {
                  "tp-id": "LTP-4",
                  "te-tp-id": 10004
                  "te": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-l2sc",
                        "encoding": "lsp-encoding-ethernet"
                      }
                    ]
                  }
                }
              ]
            }
          ]
        }
```

```
        ]
      }
    }
```

Authors' Addresses

Igor Bryskin
Huawei Technologies

EMail: Igor.Bryskin@huawei.com

Xufeng Liu
Volta Networks

EMail: xufeng.liu.ietf@gmail.com

Young Lee
Huawei Technologies

EMail: leeyoung@huawei.com