Workgroup: CCAMP Working Group
Internet-Draft:
draft-ietf-ccamp-client-signal-yang-09
Published: 10 July 2023
Intended Status: Standards Track
Expires: 11 January 2024
Authors: H. Zheng
Huawei Technologies
A. Guo
Futurewei
I. Busi
Huawei Technologies
A. Snitser
Cisco
F. Lazzeri
Ericsson

A YANG Data Model for Transport Network Client Signals

Abstract

A transport network is a server-layer network to provide connectivity services to its client. The topology and tunnel information in the transport layer has already been defined by generic Traffic-engineered models and technology-specific models (e.g., OTN, WSON). However, how the client signals are accessing to the network has not been described. These information is necessary to both client and provider.

This draft describes how the client signals are carried over transport network and defines YANG data models which are required during configuration procedure. More specifically, several client signal (of transport network) models including ETH, STM-n, FC and so on, are defined in this draft.

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 11 January 2024.

Table of Contents

1. Introduction

1.1. Overview

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic transparently across the server-layer network resources. Currently the topology and tunnel models have been defined for

transport networks, including [I-D.ietf-ccamp-otn-topo-yang] and [I-D.ietf-ccamp-otn-tunnel-model], providing server-layer topology abstraction and tunnel configuration between PEs. However, there is a missing piece for configuring how the PEs should map the client-layer traffic, received from the CE, over the server-layer tunnels: this gap is expected to be solved in this document.

This document defines a data model of all transport network client signals, using YANG language defined in [RFC7950]. The model can be used by applications exposing to a transport network controller via a RESTconf interface. Furthermore, it can be used by an application for the following purposes (but not limited to):

- To request/update an end-to-end service by driving a new tunnel to be set up to support this service;
- To request/update an end-to-end service by using an existing tunnel;
- To receive notification with regard to the information change of the given service;

The YANG modules defined in this document conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

1.2. Prefixs in Model 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, including [RFC6991], [RFC8294] and [I-D.ietf-ccamp-otn-tunnel-model], which are shown as follow.

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
te-types	ietf-te-types	[RFC8776]
rt-types	ietf-routing-types	[RFC8294]
l1-types	ietf-layer1-types	[ietf-ccamp-layer1-types]
etht-types	ietf-eth-tran-types	This Document
clntsvc	ietf-trans-client-service	This Document
ethtsvc	ietf-eth-tran-service	This Document
clntsvc-types	ietf-trans-client-svc-types	This Document

2. Terminology and Notations

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in the YANG data tree presented later in this document is defined in [RFC8340]. They are provided below for reference.

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

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

Symbols after data node names: "?" means an optional node, "!" means a presence container, and "" denotes a list and leaf-list.

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

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

3. Transport Network Client Signal Overview

3.1. Overview of Service Request and Network Configuration Scenarios

A global view of a multi-domain service can be described as the Figure 1 . The customer is usually responsible to configure the CE nodes and to request to the provider the service intent, from the CE nodes perspective, while the provider is responsible to configure the whole network (including the PE nodes) to support the customer service intent. Generally speaking, the network configurations required to support a customer service can be split into two different groups: CE-PE and PE-PE. The CE-PE configuration deals with the client layer one-hop access link, while PE-PE configuration deals with the server layer tunnel. In Figure 1 we mark the intermediate nodes as 'P', which has same switching capability of PE but just not the 'end-point'. In this example, the link P-P and PE-P are a server-layer intra-domain or inter-domain link.

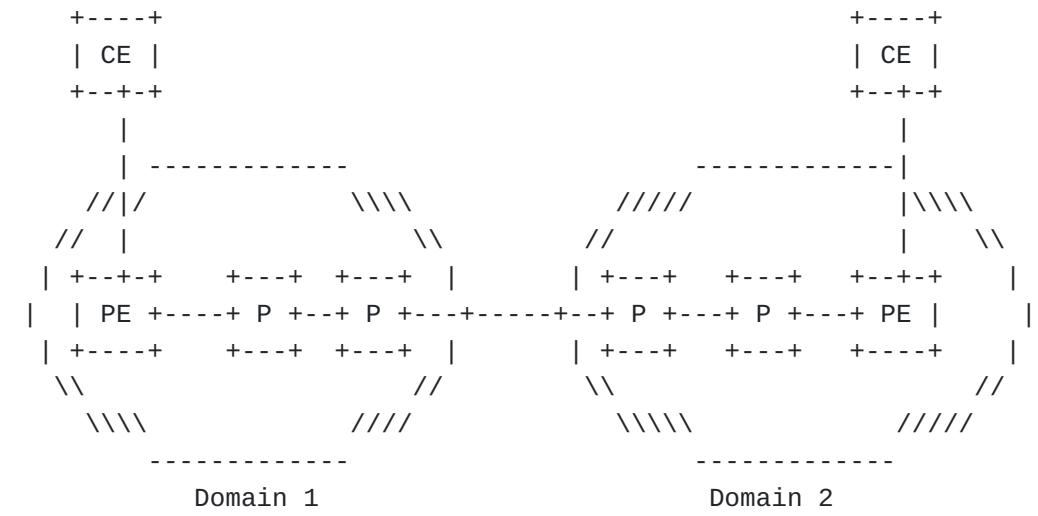

Figure 1: Global view of Client Service with the Network Provider

According to the responsibilities of each controller in [RFC8453], the controllers have different views of the service request and network configuration. The duty of CNC is to give the MDSC a description of the customer service intent: candidate YANG models include L1CSM [I-D.ietf-ccamp-l1csm-yang], L2SM [RFC8466] and L3SM [RFC8299], which are classified as customer service models, according to [RFC8309]. These models provide necessary attributes to describe the customer service intent from the customer/CE perspective, and do not provide any specific network configuration. These models also implies that the customer service description can be considered in a separate manner rather than integratig with network configurations, which also enable the controllers to abstract/virtualize the network resource to make them visible to the customer and also easier to manage. In other words, the network knowledge is not necessary at CNC and CMI, which is seen in an abstracted form as shown in Figure 2.

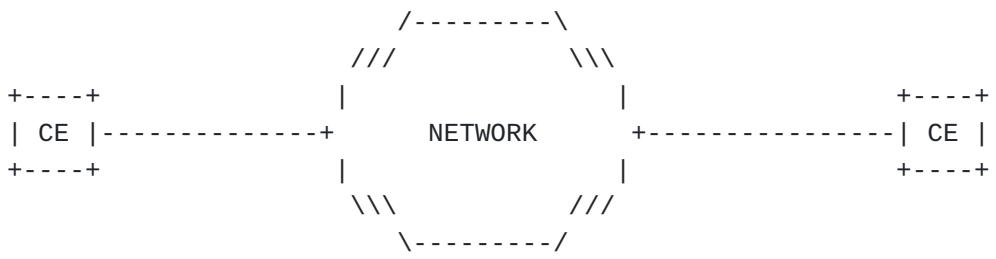

Figure 2: CNC Viewpoint on the Client Service

The functionalities of MDSC have been described in [RFC8453], which include the customer mapping/translation and multi-domain coordination. By receiving the request from CNC, MDSC need to understand what network configuration can support the customer service intent and turn to the corresponding PNCs for configuration. The service request is therefore decomposed by MDSC into a few network configurations and forwarded to one or multiple PNCs respectively in single-domain and multi-domain scenario. In general, the MDSC has the view of both PE and CE nodes and of some abstract information regarding the P nodes, as shown in Figure 3. It is worth noting that this MDSC view is different with Figure 1 at the intra-domain link. Usually these details are hidden, for scalability purposes, and therefore the MDSC has only an abstract view of each domain internal topology.

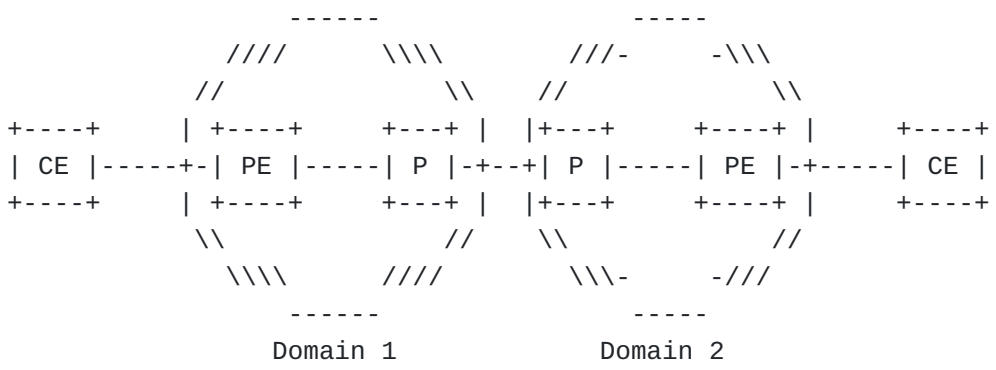

Figure 3: MDSC view of both Client Service and Network Abstraction

PNC is the controller that configure the physical devices, based on the network configuration received from the MDSC. Each PNC has the detailed view of its own domain, the example of view from PNC in domain 1 is shown in Figure 4. The PNC has all the detailed topology information on PE and P nodes and on the intra-domain links. The PNC configures the tunnel/tunnel segment within its domain based on the network configuration provided by the MDSC. The PNC also configures the network part of the CE-PE access links as well as the mapping of the client-layer traffic and the server-layer tunnels, based on the network configuration provided by the MDSC. The interaction between PNC and MDSC for the client-layer network configuration is accomplished by the models defined in this draft.

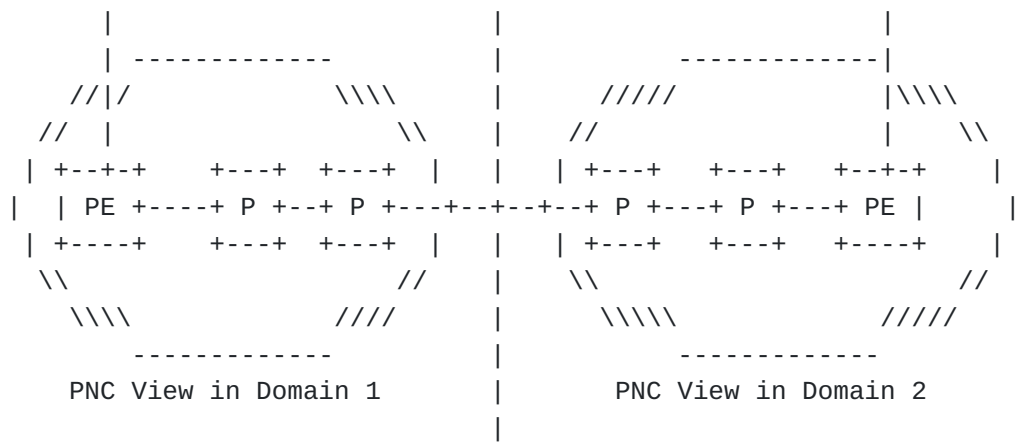

Figure 4: PNC view on Network Configuration

3.2. Applicability of Proposed Model

Existing TE and technology-specific models, such as topology models and tunnel models, support the network configuration among PEs and Ps. The customer service models, such as L1CSM, L2SM and L3SM, focus on describing the attributes among CEs. However, there is a missing piece on how to configure the CE-PE session. The models defined in this document provide the configuration on CE-PE when the provider server-layer network is TE-based technology.

In the example of OTN as the server-layer transport network, a full list of G-PID was summarized in [RFC7139], which can be divided into a few categories. The G-PID signals can be categorized into transparent and non-transparent. Examples of transparent signals may include Ethernetphysical interfaces, FC, STM-n and so on. In this approach the OTN devices is not aware of the client signal type, and this information is only necessary among the controllers. Once the OTN tunnel is set up, there is no switching requested on the client layer, and therefore only signal mapping is needed, without a client tunnel set up. The models that supporting the configuration of transparent signals are defined in Section 4.2. The other category would be non-transparent, such as Carrier Ethernet and MPLS-TP, with a switching request on the client layer. Once the OTN tunnel is set up, a corresponding tunnel in the client layer has to be set up to carry services. The models that supporting the configuration of transparent signals are defined in Section 4.1.

It is also worth noting that some client signal can be carried over multiple types of networks. For example, the Ethernet services can be carried over either OTN or Ethernet TE tunnels (over optical or microwave networks). The model specified in this document allows the support from networks with different technologies. The list of

identities for client signals are defined in [I-D.ietf-ccamp-layer1-types].

3.3. State of Services

States have been defined to retrieve the status of the delivered services. A few generic states defined in [RFC8776] are reused in this document. These states include the operational state and the provisioning state.

A few other parameters are defined for the management of the services. Given the complicated labor division, the creation, update and maintainance may have different responsible systems. So the log of the service would be helpful and should be easy to retrived in a standard way as well. These information include, but not restricted to the following:

* When the service is created and has been last updated, these are specified in the creation-time and last-updated-time.

* Who created the service and who made the last update to the service, these are specified in the created-by and last-updated-by.

* The owner of the service is specifed as owned-by, which would be useful when the service is delegated by or to a specific system. The identifier of the system is used to represent such information.

4. YANG Model for Transport Network Client Signal

4.1. YANG Tree for Ethernet Service

```
module: ietf-eth-tran-service
  +--rw etht-svc
     +--rw globals
     |  +--rw named-bandwidth-profiles* [bandwidth-profile-name]
     |     +--rw bandwidth-profile-name    string
     |     +--rw bandwidth-profile-type?
     |     |       etht-types:bandwidth-profile-type
     |     +--rw CIR?                      uint64
     |     +--rw CBS?                      uint64
     |     +--rw EIR?                      uint64
     |     +--rw EBS?                      uint64
     |     +--rw color-aware?              boolean
     |     +--rw coupling-flag?            boolean
     +--rw etht-svc-instances* [etht-svc-name]
        +--rw etht-svc-name             string
        +--rw etht-svc-title?           string
        +--rw etht-svc-descr?           string
        +--rw etht-svc-customer?        string
        +--rw etht-svc-type?            etht-types:service-type
        +--rw etht-svc-lifecycle?       etht-types:lifecycle-status
        +--rw te-topology-identifier
        |  +--rw provider-id?   te-global-id
        |  +--rw client-id?     te-global-id
        |  +--rw topology-id?   te-topology-id
        +--rw resilience
        |  +--rw protection
        |  |  +--rw enable?                          boolean
        |  |  +--rw protection-type?                 identityref
        |  |  +--rw protection-reversion-disable?    boolean
        |  |  +--rw hold-off-time?                   uint32
        |  |  +--rw wait-to-revert?                  uint16
        |  |  +--rw aps-signal-id?                   uint8
        |  +--rw restoration
        |     +--rw enable?                           boolean
        |     +--rw restoration-type?                 identityref
        |     +--rw restoration-scheme?               identityref
        |     +--rw restoration-reversion-disable?    boolean
        |     +--rw hold-off-time?                    uint32
        |     +--rw wait-to-restore?                  uint16
        |     +--rw wait-to-revert?                   uint16
        +--rw etht-svc-end-points* [etht-svc-end-point-name]
        |  +--rw etht-svc-end-point-name                  string
        |  +--rw etht-svc-end-point-id?                   string
        |  +--rw etht-svc-end-point-descr?                string
        |  +--rw topology-role?
        |  |       identityref
        |  +--rw resilience
        |  +--rw etht-svc-access-points* [access-point-id]
```

```
|  |  +--rw access-point-id    string
|  |  +--rw access-node-id?    te-types:te-node-id
|  |  +--rw access-ltp-id?     te-types:te-tp-id
|  |  +--rw access-role?       identityref
|  |  +--rw pm-config
|  |  |  +--rw pm-enable?             boolean
|  |  |  +--rw sending-rate-high?     uint64
|  |  |  +--rw sending-rate-low?      uint64
|  |  |  +--rw receiving-rate-high?   uint64
|  |  |  +--rw receiving-rate-low?    uint64
|  |  +--ro state
|  |  |  +--ro operational-state?    identityref
|  |  |  +--ro provisioning-state?   identityref
|  |  +--ro performance?       identityref
|  +--rw service-classification-type?
|  |       identityref
|  +--rw (service-classification)?
|  |  +--:(port-classification)
|  |  +--:(vlan-classification)
|  |     +--rw outer-tag!
|  |     |  +--rw tag-type?
|  |     |  |       etht-types:eth-tag-classify
|  |     |  +--rw (individual-bundling-vlan)?
|  |     |     +--:(individual-vlan)
|  |     |     |  +--rw vlan-value?   etht-types:vlanid
|  |     |     +--:(vlan-bundling)
|  |     |        +--rw vlan-range?
|  |     |                etht-types:vid-range-type
|  |     +--rw second-tag!
|  |        +--rw tag-type?
|  |        |       etht-types:eth-tag-classify
|  |        +--rw (individual-bundling-vlan)?
|  |           +--:(individual-vlan)
|  |           |  +--rw vlan-value?   etht-types:vlanid
|  |           +--:(vlan-bundling)
|  |              +--rw vlan-range?
|  |                      etht-types:vid-range-type
|  +--rw split-horizon-group?                    string
|  +--rw (direction)?
|  |  +--:(symmetrical)
|  |  |  +--rw ingress-egress-bandwidth-profile
|  |  |     +--rw (style)?
|  |  |        +--:(named)
|  |  |        |  +--rw bandwidth-profile-name?   leafref
|  |  |        +--:(value)
|  |  |           +--rw bandwidth-profile-type?
|  |  |           |       etht-types:bandwidth-profile-type
|  |  |           +--rw CIR?                     uint64
|  |  |           +--rw CBS?                     uint64
```

```
|  |  |          +--rw EIR?                      uint64
|  |  |          +--rw EBS?                      uint64
|  |  |          +--rw color-aware?              boolean
|  |  |          +--rw coupling-flag?            boolean
|  |  +--:(asymmetrical)
|  |     +--rw ingress-bandwidth-profile
|  |     |  +--rw (style)?
|  |     |     +--:(named)
|  |     |     |  +--rw bandwidth-profile-name?   leafref
|  |     |     +--:(value)
|  |     |        +--rw bandwidth-profile-type?
|  |     |        |       etht-types:bandwidth-profile-type
|  |     |        +--rw CIR?                      uint64
|  |     |        +--rw CBS?                      uint64
|  |     |        +--rw EIR?                      uint64
|  |     |        +--rw EBS?                      uint64
|  |     |        +--rw color-aware?              boolean
|  |     |        +--rw coupling-flag?            boolean
|  |     +--rw egress-bandwidth-profile
|  |        +--rw (style)?
|  |           +--:(named)
|  |           |  +--rw bandwidth-profile-name?   leafref
|  |           +--:(value)
|  |              +--rw bandwidth-profile-type?
|  |              |       etht-types:bandwidth-profile-type
|  |              +--rw CIR?                      uint64
|  |              +--rw CBS?                      uint64
|  |              +--rw EIR?                      uint64
|  |              +--rw EBS?                      uint64
|  |              +--rw color-aware?              boolean
|  |              +--rw coupling-flag?            boolean
|  +--rw vlan-operations
|     +--rw (direction)?
|        +--:(symmetrical)
|        |  +--rw symmetrical-operation
|        |     +--rw pop-tags?    uint8
|        |     +--rw push-tags
|        |        +--rw outer-tag!
|        |        |  +--rw tag-type?
|        |        |  |       etht-types:eth-tag-type
|        |        |  +--rw vlan-value?    etht-types:vlanid
|        |        |  +--rw default-pcp?   uint8
|        |        +--rw second-tag!
|        |           +--rw tag-type?
|        |           |       etht-types:eth-tag-type
|        |           +--rw vlan-value?    etht-types:vlanid
|        |           +--rw default-pcp?   uint8
|        +--:(asymmetrical)
|           +--rw asymmetrical-operation
```

```
|              +--rw ingress
|              |  +--rw pop-tags?    uint8
|              |  +--rw push-tags
|              |     +--rw outer-tag!
|              |     |  +--rw tag-type?
|              |     |  |       etht-types:eth-tag-type
|              |     |  +--rw vlan-value?
|              |     |  |       etht-types:vlanid
|              |     |  +--rw default-pcp?   uint8
|              |     +--rw second-tag!
|              |        +--rw tag-type?
|              |        |       etht-types:eth-tag-type
|              |        +--rw vlan-value?
|              |        |       etht-types:vlanid
|              |        +--rw default-pcp?   uint8
|              +--rw egress
|                 +--rw pop-tags?    uint8
|                 +--rw push-tags
|                    +--rw outer-tag!
|                    |  +--rw tag-type?
|                    |  |       etht-types:eth-tag-type
|                    |  +--rw vlan-value?
|                    |  |       etht-types:vlanid
|                    |  +--rw default-pcp?   uint8
|                    +--rw second-tag!
|                       +--rw tag-type?
|                       |       etht-types:eth-tag-type
|                       +--rw vlan-value?
|                       |       etht-types:vlanid
|                       +--rw default-pcp?   uint8
+--rw underlay
|  +--rw (technology)?
|  |  +--:(native-ethernet)
|  |  |  +--rw eth-tunnels* [name]
|  |  |     +--rw name
|  |  |     |       -> /te:te/tunnels/tunnel/name
|  |  |     +--rw encoding?         identityref
|  |  |     +--rw switching-type?   identityref
|  |  +--:(frame-base)
|  |  |  +--rw otn-tunnels* [name]
|  |  |     +--rw name
|  |  |     |       -> /te:te/tunnels/tunnel/name
|  |  |     +--rw encoding?         identityref
|  |  |     +--rw switching-type?   identityref
|  |  +--:(mpls-tp)
|  |     +--rw pw
|  |        +--rw pw-id?                      string
|  |        +--rw pw-name?                    string
|  |        +--rw transmit-label?
```

```
|  |        |       rt-types:mpls-label
|  |        +--rw receive-label?
|  |        |       rt-types:mpls-label
|  |        +--rw encapsulation-type?          identityref
|  |        +--ro oper-status?                 identityref
|  |        +--rw ingress-bandwidth-profile
|  |        |  +--rw (style)?
|  |        |     +--:(named)
|  |        |     |  +--rw bandwidth-profile-name?   leafref
|  |        |     +--:(value)
|  |        |        +--rw bandwidth-profile-type?
|  |        |        |       etht-types:bandwidth-profile-type
|  |        |        +--rw CIR?                      uint64
|  |        |        +--rw CBS?                      uint64
|  |        |        +--rw EIR?                      uint64
|  |        |        +--rw EBS?                      uint64
|  |        +--rw pw-paths* [path-id]
|  |           +--rw path-id       uint8
|  |           +--rw tp-tunnels* [name]
|  |              +--rw name    string
|  +--rw src-split-horizon-group?   string
|  +--rw dst-split-horizon-group?   string
+--rw admin-status?             identityref
+--ro state
   +--ro operational-state?    identityref
   +--ro provisioning-state?   identityref
   +--ro creation-time?        yang:date-and-time
   +--ro last-updated-time?    yang:date-and-time
   +--ro created-by?           string
   +--ro last-updated-by?      string
   +--ro owned-by?             string
```

4.2. YANG Tree for other Transport Network Client Signal Model

```
module: ietf-trans-client-service
  +--rw client-svc
     +--rw client-svc-instances* [client-svc-name]
        +--rw client-svc-name           string
        +--rw client-svc-title?         string
        +--rw client-svc-descr?         string
        +--rw client-svc-customer?      string
        +--rw resilience
        +--rw te-topology-identifier
        |  +--rw provider-id?   te-global-id
        |  +--rw client-id?     te-global-id
        |  +--rw topology-id?   te-topology-id
        +--rw admin-status?             identityref
        +--rw src-access-ports
        |  +--rw access-node-id?   te-types:te-node-id
        |  +--rw access-ltp-id?    te-types:te-tp-id
        |  +--rw client-signal?    identityref
        +--rw dst-access-ports
        |  +--rw access-node-id?   te-types:te-node-id
        |  +--rw access-ltp-id?    te-types:te-tp-id
        |  +--rw client-signal?    identityref
        +--rw direction?                identityref
        +--rw svc-tunnels* [tunnel-name]
        |  +--rw tunnel-name    string
        +--ro operational-state?        identityref
        +--ro provisioning-state?       identityref
        +--ro creation-time?            yang:date-and-time
        +--ro last-updated-time?        yang:date-and-time
        +--ro created-by?               string
        +--ro last-updated-by?          string
        +--ro owned-by?                 string
```

5. YANG Code for Transport Network Client Signal

5.1. The ETH Service YANG Code

This module imports typedefs and modules from [RFC6991], [RFC8294], [RFC8776].

```
<CODE BEGINS> file "ietf-eth-tran-service@2021-01-11.yang"

module ietf-eth-tran-service {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-eth-tran-service";

  prefix "ethtsvc";

  import ietf-yang-types {
    prefix "yang";
    reference "RFC 6991 - Common YANG Data Types";
  }

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

  import ietf-eth-tran-types {
    prefix "etht-types";
    reference "RFC XXXX - A YANG Data Model for Transport
               Network Client Signals";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference "RFC 8294 - Common YANG Data Types for the
               Routing Area";

  }

  import ietf-te {
    prefix "te";
    reference "RFC YYYY - A YANG Data Model for Traffic
               Engineering Tunnels and Interfaces";
  }

  organization
    "Internet Engineering Task Force (IETF) CCAMP WG";
  contact
    "
      WG List: <mailto:ccamp@ietf.org>

      ID-draft editor:
        Haomian Zheng (zhenghaomian@huawei.com);
        Italo Busi (italo.busi@huawei.com);
        Aihua Guo (aihuaguo.ietf@gmail.com);
        Anton Snitser (asnizar@cisco.com);
        Francesco Lazzeri (francesco.lazzeri@ericsson.com);
    ";
```

```
description
  "This module defines a YANG data model for describing
   the Ethernet services. The model fully conforms to the
   Network Management Datastore Architecture (NMDA).

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

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).
   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

revision 2021-01-11 {
  description
    "version -04 as an WG document";
  reference
    "draft-ietf-ccamp-client-signal-yang";
}

/*
 * Groupings
 */

grouping vlan-classification {
  description
    "A grouping which represents classification
    on an 802.1Q VLAN tag.";

  leaf tag-type {
    type etht-types:eth-tag-classify;
    description
      "The tag type used for VLAN classification.";
  }
  choice individual-bundling-vlan {
    description
      "VLAN based classification can be individual
       or bundling.";

    case individual-vlan {
      leaf vlan-value {
        type etht-types:vlanid;
        description
          "VLAN ID value.";
      }
```

```
      }

      case vlan-bundling {
        leaf vlan-range {
          type etht-types:vid-range-type;
          description
            "List of VLAN ID values.";
        }
      }
    }
  }

  grouping vlan-write {
    description
      "A grouping which represents push/pop operations
       of an 802.1Q VLAN tag.";

    leaf tag-type {
      type etht-types:eth-tag-type;
      description
        "The VLAN tag type to push/swap.";
    }
    leaf vlan-value {
      type etht-types:vlanid;
      description
        "The VLAN ID value to push/swap.";
    }
/*
 * To be added: this attribute is used when:
 * a) the ETH service has only one CoS (as in current version)
 * b) as a default when a mapping between a given CoS value
 *    and the PCP value is not defined (in future versions)
 */
    leaf default-pcp {
      type uint8 {
        range "0..7";
      }
      description
        "The default Priority Code Point (PCP) value to push/swap";
    }
  }

  grouping vlan-operations {
    description
      "A grouping which represents VLAN operations.";

      leaf pop-tags {
      type uint8 {
        range "1..2";
```

```
  }
  description
    "The number of VLAN tags to pop (or swap if used in
     conjunction with push-tags)";
}
container push-tags {
  description
    "The VLAN tags to push (or swap if used in
     conjunction with pop-tags)";

  container outer-tag {
    presence
      "Indicates existence of the outermost VLAN tag to
       push/swap";

    description
      "The outermost VLAN tag to push/swap.";

    uses vlan-write;
  }
  container second-tag {
  must
    '../outer-tag/tag-type = "etht-types:s-vlan-tag-type" and ' +
    'tag-type = "etht-types:c-vlan-tag-type"'
    {

      error-message
        "
          When pushing/swapping two tags, the outermost tag must
          be specified and of S-VLAN type and the second
          outermost tag must be of C-VLAN tag type.
        ";
      description
        "
          For IEEE 802.1Q interoperability, when pushing/swapping
          two tags, it is required that the outermost tag exists
          and is an S-VLAN, and the second outermost tag is a
          C-VLAN.
        ";
    }

    presence
      "Indicates existence of a second outermost VLAN tag to
       push/swap";

    description
      "The second outermost VLAN tag to push/swap.";

    uses vlan-write;
```

```
    }
  }
}

grouping named-or-value-bandwidth-profile {
  description
    "A grouping to configure a bandwdith profile either by
     referencing a named bandwidth profile or by
     configuring the values of the bandwidth profile attributes.";
  choice style {
    description
      "Whether the bandwidth profile is named or defined by value";

    case named {
      description
        "Named bandwidth profile.";
      leaf bandwidth-profile-name {
        type leafref {
           path "/ethtsvc:etht-svc/ethtsvc:globals/"
           + "ethtsvc:named-bandwidth-profiles/"
           + "ethtsvc:bandwidth-profile-name";
           }
        description
          "Name of the bandwidth profile.";
      }
    }
    case value {
      description
        "Bandwidth profile configured by value.";
      uses etht-types:etht-bandwidth-profiles;
    }
  }
}

grouping bandwidth-profiles {
  description
    "A grouping which represent bandwidth profile configuration.";

  choice direction {
    description
      "Whether the bandwidth profiles are symmetrical or
       asymmetrical";
    case symmetrical {
      description
        "The same bandwidth profile is used to describe both
         the ingress and the egress bandwidth profile.";
      container ingress-egress-bandwidth-profile {
        description
          "The bandwdith profile used in both directions.";
```

```
        uses named-or-value-bandwidth-profile;
      }
    }
    case asymmetrical {
      description
        "Ingress and egress bandwidth profiles can be specified.";
      container ingress-bandwidth-profile {
        description
          "The bandwdith profile used in the ingress direction.";
        uses named-or-value-bandwidth-profile;
      }
      container egress-bandwidth-profile {
        description
          "The bandwdith profile used in the egress direction.";
        uses named-or-value-bandwidth-profile;
      }
    }
  }
}

grouping etht-svc-access-parameters {
  description
    "ETH services access parameters";

  leaf access-node-id {
    type te-types:te-node-id;
    description
      "The identifier of the access node in
       the ETH topology.";
  }
  leaf access-ltp-id {
    type te-types:te-tp-id;
    description
      "The TE link termination point identifier, used
       together with access-node-id to identify the
       access LTP.";
  }
  leaf access-role {
    type identityref {
      base etht-types:access-role;
    }
    description
      "Indicate the role of access, e.g., working or protection. ";
  }

  container pm-config {
    uses pm-config-grouping;
    description
    "This grouping is used to set the threshold value for
```

```
    performance monitoring. ";
  }

  container state {
    config false;
    description
    "The state is used to monitor the status of service. ";
    leaf operational-state {
      type identityref {
        base te-types:tunnel-state-type;
      }
      description
      "Indicating the operational state of client signal. ";
    }
    leaf provisioning-state {
      type identityref {
        base te-types:lsp-state-type;
      }
      description
      "Indicating the provisional state of client signal,
      especially when there is a change, i.e., revise, create. ";
    }
  }

  leaf performance {
    type identityref {
      base etht-types:performance;
    }
    config false;
    description
      "Performance Monitoring for the service. ";
  }


}

grouping etht-svc-tunnel-parameters {
  description
    "ETH services tunnel parameters.";
  choice technology {
    description
      "Service multiplexing is optional and flexible.";

    case native-ethernet {
      /*
       placeholder to support proprietary multiplexing
       (for further discussion)
      */
      list eth-tunnels {
```

```
          key name;
          description
            "ETH Tunnel list in native Ethernet scenario.";
          uses tunnels-grouping;
        }
      }

      case frame-base {
        list otn-tunnels {
          key name;
          description
            "OTN Tunnel list in Frame-based scenario.";
          uses tunnels-grouping;
        }
      }

      case mpls-tp {
        container pw {
          description
            "Pseudowire information for Ethernet over MPLS-TP.";
          uses pw-segment-grouping;
        }
      }
    }

/*
 * Open issue: can we constraints it to be used only with mp services?
 */
    leaf src-split-horizon-group {
      type string;
      description
      "Identify a split horizon group at the Tunnel source TTP";
    }
    leaf dst-split-horizon-group {
      type string;
      description
      "Identify a split horizon group at the Tunnel destination TTP";
    }
  }

  grouping  etht-svc-pm-threshold-config {
    description
      "Configuraiton parameters for Ethernet service PM thresholds.";

    leaf sending-rate-high {
      type uint64;
      description
        "High threshold of packet sending rate in kbps.";
    }
```

```
  leaf sending-rate-low {
    type uint64;
    description
      "Low threshold of packet sending rate in kbps.";
  }
  leaf receiving-rate-high {
    type uint64;
    description
      "High threshold of packet receiving rate in kbps.";
  }
  leaf receiving-rate-low {
    type uint64;
    description
      "Low threshold of packet receiving rate in kbps.";
  }
}

grouping  etht-svc-pm-stats {
  description
    "Ethernet service PM statistics.";

  leaf sending-rate-too-high {
    type uint32;
    description
      "Counter that indicates the number of times the
      sending rate is above the high threshold";
  }
  leaf sending-rate-too-low {
    type uint32;
    description
      "Counter that indicates the number of times the
      sending rate is below the low threshold";
  }
  leaf receiving-rate-too-high {
    type uint32;
    description
      "Counter that indicates the number of times the
      receiving rate is above the high threshold";
  }
  leaf receiving-rate-too-low {
    type uint32;
    description
      "Counter that indicates the number of times the
      receiving rate is below the low threshold";
  }
}

grouping  etht-svc-instance-config {
  description
```

```
  "Configuraiton parameters for Ethernet services.";

leaf etht-svc-name {
  type string;
  description
    "Name of the ETH service.";
}

leaf etht-svc-title {
  type string;
  description
    "The Identifier of the ETH service.";
}

leaf etht-svc-descr {
  type string;
  description
    "Description of the ETH service.";
}

leaf etht-svc-customer {
  type string;
  description
    "Customer of the ETH service.";
}

 leaf etht-svc-type {
  type etht-types:service-type;
  description
    "Type of ETH service (p2p, mp2mp or rmp).";
  /* Add default as p2p */
}

 leaf etht-svc-lifecycle {
  type etht-types:lifecycle-status;
  description
    "Lifecycle state of ETH service.";
  /* Add default as installed  */
}
uses te-types:te-topology-identifier;

uses resilience-grouping;

list etht-svc-end-points {
  key etht-svc-end-point-name;
  description
    "The logical end point for the ETH service. ";
  uses etht-svc-end-point-grouping;
}
```

```
  container underlay {
    description
      "The unterlay tunnel information that carrying the
      ETH service. ";
    uses etht-svc-tunnel-parameters;
  }

  leaf admin-status {
    type identityref {
      base te-types:tunnel-admin-state-type;
    }
    default te-types:tunnel-admin-state-up;
    description "ETH service administrative state.";
  }
}

grouping  etht-svc-instance-state {
  description
    "State parameters for Ethernet services.";

  leaf operational-state {
  type identityref {
      base te-types:tunnel-state-type;
    }
    default te-types:tunnel-state-up;
  description "ETH service operational state.";
  }
  leaf provisioning-state {
    type identityref {
      base te-types:lsp-state-type;
    }
    description "ETH service provisioning state.";
  }
  leaf creation-time {
    type yang:date-and-time;
    description
      "Time of ETH service creation.";
  }
  leaf last-updated-time {
    type yang:date-and-time;
    description
      "Time of ETH service last update.";
  }

  leaf created-by {
     type string;
     description
       "The client signal is created by whom,
```

```
        can be a system or staff ID.";
    }
    leaf last-updated-by {
      type string;
      description
        "The client signal is last updated by whom,
         can be a system or staff ID.";
    }
    leaf owned-by {
      type string;
      description
        "The client signal is last updated by whom,
         can be a system ID.";
    }
}

/*
 * Data nodes
 */

container etht-svc {
  description
    "ETH services.";

  container globals {
    description
      "Globals Ethernet configuration data container";
    list named-bandwidth-profiles {
      key bandwidth-profile-name;
      description
        "List of named bandwidth profiles used by
         Ethernet services.";

      leaf bandwidth-profile-name {
        type string;
        description
          "Name of the bandwidth profile.";
      }
      uses etht-types:etht-bandwidth-profiles;
    }
  }

  list etht-svc-instances {
    key etht-svc-name;
    description
      "The list of p2p ETH service instances";

    uses etht-svc-instance-config;
```

```
    container state {
      config false;
      description
        "Ethernet Service states.";

      uses etht-svc-instance-state;
    }
  }
}

grouping resilience-grouping {
  description
    "Grouping for resilience configuration. ";
  container resilience {
  description
    "To configure the data plane protection parameters,
    currently a placeholder only, future candidate attributes
    include, Revert, WTR, Hold-off Timer, ...";
    uses te:protection-restoration-properties;
  }
}

grouping etht-svc-end-point-grouping {
  description
    "Grouping for the end point configuration.";
  leaf etht-svc-end-point-name {
    type string;
    description
    "The name of the logical end point of ETH service. ";
  }

  leaf etht-svc-end-point-id {
    type string;
    description
    "The identifier of the logical end point of ETH service.";
  }

  leaf etht-svc-end-point-descr {
    type string;
    description
    "The description of the logical end point of ETH service. ";
  }

  leaf topology-role {
    type identityref {
      base etht-types:topology-role;
    }
    description
    "Indicating the underlay topology role,
```

```
      e.g., hub,spoke, any-to-any ";
    }

    container resilience {
      description
      "Placeholder for resilience configuration, for future study. ";
    }

    list etht-svc-access-points {
      key access-point-id;
      min-elements "1";
/*
  Open Issue:
    Is it possible to limit the max-elements only for p2p services?
      max-elements "2";
*/
      description
        "List of the ETH trasport services access point instances.";

      leaf access-point-id {
        type string;
        description
          "ID of the service access point instance";
      }
      uses etht-svc-access-parameters;
    }

    leaf service-classification-type {
      type identityref {
        base etht-types:service-classification-type;
      }
      description
        "Service classification type.";
    }

    choice service-classification {
      description
        "Access classification can be port-based or
         VLAN based.";

      case port-classification {
        /* no additional information */
      }

      case vlan-classification {
        container outer-tag {
          presence "The outermost VLAN tag exists";
          description
            "Classifies traffic using the outermost VLAN tag.";
```

```
          uses vlan-classification;
        }
        container second-tag {
          must
            '../outer-tag/tag-type = "etht-types:classify-s-vlan" and '
            'tag-type = "etht-types:classify-c-vlan"'
          {
            error-message
              "
                When matching two tags, the outermost tag must be
                specified and of S-VLAN type and the second
                outermost tag must be of C-VLAN tag type.
              ";
            description
              "
                For IEEE 802.1Q interoperability, when matching two
                tags, it is required that the outermost tag exists
                and is an S-VLAN, and the second outermost tag is a
                C-VLAN.
              ";
          }
          presence "The second outermost VLAN tag exists";

          description
            "Classifies traffic using the second outermost VLAN tag.";

          uses vlan-classification;
        }
      }
    }

/*
 * Open issue: can we constraints it to be used only with mp services?
 */
    leaf split-horizon-group {
      type string;
      description "Identify a split horizon group";
    }

    uses bandwidth-profiles;

    container vlan-operations {
      description
        "Configuration of VLAN operations.";
      choice direction {
        description
          "Whether the VLAN operations are symmetrical or
           asymmetrical";
        case symmetrical {
```

```
        container symmetrical-operation {
          uses vlan-operations;
          description
            "Symmetrical operations.
             Expressed in the ingress direction, but
             the reverse operation is applied to egress traffic";
        }
      }
      case asymmetrical {
        container asymmetrical-operation {
          description "Asymmetrical operations";
          container ingress {
            uses vlan-operations;
            description "Ingress operations";
          }
          container egress {
            uses vlan-operations;
            description "Egress operations";
          }
        }
      }
    }
  }
}

grouping pm-config-grouping {
  description
    "Grouping used for Performance Monitoring Configuration. ";
  leaf pm-enable {
    type boolean;
    description
    "Whether to enable the performance monitoring.";
  }

  leaf sending-rate-high {
    type uint64;
    description
    "The upperbound of sending rate.";
  }

  leaf sending-rate-low {
    type uint64;
    description
    "The lowerbound of sending rate.";
  }

  leaf receiving-rate-high {
    type uint64;
    description
```

```
    "The upperbound of receiving rate.";
  }

  leaf receiving-rate-low {
    type uint64;
    description
    "The lowerbound of receiving rate.";
  }
}

grouping pw-segment-grouping {
  description
    "Grouping used for PW configuration. ";
  leaf pw-id {
    type string;
    description
    "The Identifier information of pseudowire. ";
  }

  leaf pw-name {
    type string;
    description
    "The name information of pseudowire.";
  }

  leaf transmit-label {
    type rt-types:mpls-label;
    description
    "Transmit label information in PW. ";
  }

  leaf receive-label {
    type rt-types:mpls-label;
    description
    "Receive label information in PW. ";
  }

  leaf encapsulation-type {
    type identityref {
      base etht-types:encapsulation-type;
    }
    description
      "The encapsulation type, raw or tag. ";
  }

  leaf oper-status {
    type identityref {
      base te-types:tunnel-state-type;
    }
```

```
    config false;
    description
      "The operational state of the PW segment. ";
  }

  container ingress-bandwidth-profile {
    description
      "Bandwidth Profile for ingress. ";
    uses pw-segment-named-or-value-bandwidth-profile;
  }

  list pw-paths {
    key path-id;
    description
      "A list of pw paths. ";

    leaf path-id {
      type uint8;
      description
        "The identifier of pw paths. ";

    }

    list tp-tunnels {
      key name;
      description
        "Names of TP Tunnel underlay";
      leaf name {
        type string;
        description
          "Names of TP Tunnel underlay";
        }
    }
  }

}

grouping pw-segment-named-or-value-bandwidth-profile {
  description
    "A grouping to configure a bandwdith profile either by
     referencing a named bandwidth profile or by
     configuring the values of the bandwidth profile attributes.";
  choice style {
    description
      "Whether the bandwidth profile is named or defined by value";
    case named {
      description
        "Named bandwidth profile.";
      leaf bandwidth-profile-name {
```

```
          type leafref {
            path "/ethtsvc:etht-svc/ethtsvc:globals/"
            + "ethtsvc:named-bandwidth-profiles/"
            + "ethtsvc:bandwidth-profile-name";
          }
          description
            "Name of the bandwidth profile.";
        }
      }
      case value {
        description
          "Bandwidth profile configured by value.";
        uses etht-types:pw-segement-bandwidth-profile-grouping;
      }
    }
  }

  grouping tunnels-grouping {
    description
      "A group of tunnels. ";
    leaf name {
      type leafref {
        path "/te:te/te:tunnels/te:tunnel/te:name";
        require-instance false;
      }
      description "Dependency tunnel name";
    }
    leaf encoding {
      type identityref {
        base te-types:lsp-encoding-types;
      }
      description "LSP encoding type";
      reference "RFC3945";
    }
    leaf switching-type {
      type identityref {
        base te-types:switching-capabilities;
      }
      description "LSP switching type";
      reference "RFC3945";
    }
  }
}

<CODE ENDS>
```

5.2. YANG Code for ETH type

This module references a few documents including [RFC2697], [RFC2698], [RFC4115], [IEEE802.1ad], [IEEE802.1q] and [MEF10].

```
<CODE BEGINS> file "ietf-eth-tran-types@2021-07-07.yang"

module ietf-eth-tran-types {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-eth-tran-types";

  prefix "etht-types";

  organization
    "Internet Engineering Task Force (IETF) CCAMP WG";
  contact
    "
      WG List: <mailto:ccamp@ietf.org>

      ID-draft editor:
        Haomian Zheng (zhenghaomian@huawei.com);
        Italo Busi (italo.busi@huawei.com);
        Aihua Guo (aihuaguo.ietf@gmail.com);
        Anton Snitser (asnizar@cisco.com);
        Francesco Lazzeri (francesco.lazzeri@ericsson.com);
    ";

  description
    "This module defines the ETH types.
     The model fully conforms to the Network Management
     Datastore Architecture (NMDA).

     Copyright (c) 2019 IETF Trust and the persons
     identified as authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (https://trustee.ietf.org/license-info).
     This version of this YANG module is part of RFC XXXX; see
     the RFC itself for full legal notices.";

  revision 2021-07-07 {
    description
      "version -05 as a WG draft";
    reference
      "draft-ietf-ccamp-client-signal-yang";
  }

  /*
   * Identities
   */
```

```
identity eth-vlan-tag-type {
  description
    "ETH VLAN tag type.";
}

identity c-vlan-tag-type {
  base eth-vlan-tag-type;
  description
    "802.1Q Customer VLAN";
}

identity s-vlan-tag-type {
  base eth-vlan-tag-type;
  description
    "802.1Q Service VLAN (QinQ)";
}

identity service-classification-type {
  description
    "Service classification.";
}

identity port-classification {
  base service-classification-type;
  description
    "Port classification.";
}

identity vlan-classification {
  base service-classification-type;
  description
    "VLAN classification.";
}

identity eth-vlan-tag-classify {
  description
    "VLAN tag classification.";
}

identity classify-c-vlan {
  base eth-vlan-tag-classify;
  description
    "Classify 802.1Q Customer VLAN tag.
     Only C-tag type is accepted";
}

identity classify-s-vlan {
  base eth-vlan-tag-classify;
  description
    "Classify 802.1Q Service VLAN (QinQ) tag.
```

```
     Only S-tag type is accepted";
}

identity classify-s-or-c-vlan {
  base eth-vlan-tag-classify;
  description
    "Classify S-VLAN or C-VLAN tag-classify.
     Either tag is accepted";
}

identity bandwidth-profile-type {
  description
    "Bandwidth Profile Types";
}

identity mef-10-bwp {
  base bandwidth-profile-type;
  description
    "MEF 10 Bandwidth Profile";
}

identity rfc-2697-bwp {
  base bandwidth-profile-type;
  description
    "RFC 2697 Bandwidth Profile";
}

identity rfc-2698-bwp {
  base bandwidth-profile-type;
  description
    "RFC 2698 Bandwidth Profile";
}

identity rfc-4115-bwp {
  base bandwidth-profile-type;
  description
    "RFC 4115 Bandwidth Profile";
}

identity service-type {
  description
    "Type of Ethernet service.";
}

identity p2p-svc {
  base service-type;
  description
    "Ethernet point-to-point service (EPL, EVPL).";
}
```

```
identity rmp-svc {
  base service-type;
  description
    "Ethernet rooted-multitpoint service (E-TREE, EP-TREE).";
}

identity mp2mp-svc {
  base service-type;
  description
    "Ethernet multipoint-to-multitpoint service (E-LAN, EP-LAN).";
}

identity lifecycle-status {
  description
    "Lifecycle Status.";
}

identity installed {
  base lifecycle-status;
  description
    "Installed.";
}

identity planned {
  base lifecycle-status;
  description
    "Planned.";
}

identity pending-removal {
  base lifecycle-status;
  description
    "Pending Removal.";
}

/*
 * Type Definitions
 */

typedef eth-tag-type {
  type identityref {
    base eth-vlan-tag-type;
  }
  description
    "Identifies a specific ETH VLAN tag type.";
}

typedef eth-tag-classify {
  type identityref {
    base eth-vlan-tag-classify;
```

```
  }
  description
    "Identifies a specific VLAN tag classification.";
}

typedef vlanid {
  type uint16 {
    range "1..4094";
  }
  description
    "The 12-bit VLAN-ID used in the VLAN Tag header.";
}

typedef vid-range-type {
  type string {
    pattern "([1-9][0-9]{0,3}(-[1-9][0-9]{0,3})?" +
            "(,[1-9][0-9]{0,3}(-[1-9][0-9]{0,3})?)*)";
  }
  description
    "A list of VLAN Ids, or non overlapping VLAN ranges, in
     ascending order, between 1 and 4094.
     This type is used to match an ordered list of VLAN Ids, or
     contiguous ranges of VLAN Ids. Valid VLAN Ids must be in the
     range 1 to 4094, and included in the list in non overlapping
     ascending order.

     For example: 1,10-100,50,500-1000";
}

typedef bandwidth-profile-type {
  type identityref {
    base bandwidth-profile-type;
  }
  description
    "Identifies a specific Bandwidth Profile type.";
}

typedef service-type {
  type identityref {
    base service-type;
  }
  description
    "Identifies the type of Ethernet service.";
}

typedef lifecycle-status {
  type identityref {
    base lifecycle-status;
  }
```

```
  description
    "Identifies the lLifecycle Status .";
}

/*
 * Grouping Definitions
 */

grouping etht-bandwidth-profiles {
  description
    "Bandwidth profile configuration paramters.";

  leaf bandwidth-profile-type {
    type etht-types:bandwidth-profile-type;
    description
      "The type of bandwidth profile.";
  }
  leaf CIR {
    type uint64;
    description
      "Committed Information Rate in Kbps";
  }
  leaf CBS {
    type uint64;
    description
      "Committed Burst Size in in KBytes";
  }
  leaf EIR {
    type uint64;
    /* Need to indicate that EIR is not supported by RFC 2697

    must
      '../bw-profile-type = "mef-10-bwp" or ' +
      '../bw-profile-type = "rfc-2698-bwp" or ' +
      '../bw-profile-type = "rfc-4115-bwp"'

    must
      '../bw-profile-type != "rfc-2697-bwp"'
    */
    description
      "Excess Information Rate in Kbps
       In case of RFC 2698, PIR = CIR + EIR";
  }
  leaf EBS {
    type uint64;
    description
      "Excess Burst Size in KBytes.
        In case of RFC 2698, PBS = CBS + EBS";
  }
```

```
  leaf color-aware {
    type boolean;
    description
      "Indicates weather the color-mode is
      color-aware or color-blind.";
  }
  leaf coupling-flag {
    type boolean;
    /* Need to indicate that Coupling Flag is defined only for MEF 10

    must
      '../bw-profile-type = "mef-10-bwp"'
    */
    description
      "Coupling Flag.";
  }
}

identity topology-role {
  description
    "The role of underlay topology: e.g., hub, spoke,
    any-to-any.";
}

identity resilience {
  description
  "Placeholder for resilience information in data plane,
  for future study. ";
}

identity access-role {
  description
  "Indicating whether the access is a working or protection access.";
}

identity root-primary {
  base access-role;
  description
    "Designates the primary root UNI of an E-Tree service, and may als
         designates the UNI access role of E-LINE and E-LAN service.";
}

identity root-backup {
  base access-role;
  description
    "Designates the backup root UNI of an E-Tree service.";
}

identity leaf-access {
```

```
  base access-role;
  description
    "Designates the leaf UNI of an E-Tree service.";
}

identity performance {
  description
  "Placeholder for performance information, for future study.";
}

identity encapsulation-type {
  description
  "Indicating how the service is encapsulated (to PW), e.g, raw or tag
}

grouping pw-segement-bandwidth-profile-grouping {
  description
    "bandwidth profile grouping for PW segment. ";
  leaf bandwidth-profile-type {
    type etht-types:bandwidth-profile-type;
    description
      "The type of bandwidth profile.";
  }
  leaf CIR {
    type uint64;
    description
      "Committed Information Rate in Kbps";
  }
  leaf CBS {
    type uint64;
    description
      "Committed Burst Size in in KBytes";
  }
  leaf EIR {
    type uint64;
    /* Need to indicate that EIR is not supported by RFC 2697

    must
      '../bw-profile-type = "mef-10-bwp" or ' +
      '../bw-profile-type = "rfc-2698-bwp" or ' +
      '../bw-profile-type = "rfc-4115-bwp"'

    must
      '../bw-profile-type != "rfc-2697-bwp"'
    */
    description
      "Excess Information Rate in Kbps
       In case of RFC 2698, PIR = CIR + EIR";
  }
```

```
  leaf EBS {
    type uint64;
    description
      "Excess Burst Size in KBytes.
        In case of RFC 2698, PBS = CBS + EBS";
  }
}
    grouping eth-bandwidth {
    description
      "Available bandwith for ethernet.";
    leaf eth-bandwidth {
      type uint64{
        range "0..10000000000";
      }
      units "Kbps";
      description
        "Available bandwith value expressed in kilobits per second";
    }
  }

  grouping eth-label-restriction {
    description
      "Label Restriction for ethernet.";
    leaf tag-type {
      type etht-types:eth-tag-type;
      description "VLAN tag type.";
    }
    leaf priority {
      type uint8;
      description "priority.";
    }
  }
  grouping eth-label {
    description
      "Label for ethernet.";
    leaf vlanid {
      type etht-types:vlanid;
        description
          "VLAN tag id.";
    }
  }

  grouping eth-label-step {
    description "Label step for Ethernet VLAN";
    leaf eth-step {
      type uint16 {
        range "1..4095";
    }
    default 1;
```

```
      description
        "Label step which represent possible increments for
         an Ethernet VLAN tag.";
      reference
        "IEEE 802.1ad: Provider Bridges.";
    }
  }
}
```

<CODE ENDS>

5.3. Other Client Signal YANG Code

This module imports typedefs and modules from [RFC6991], [I-D.ietf-ccamp-otn-tunnel-model], [RFC8776].

```
<CODE BEGINS> file "ietf-trans-client-service@2021-01-11.yang"

 module ietf-trans-client-service {
   /* TODO: FIXME */
     yang-version 1.1;

   namespace "urn:ietf:params:xml:ns:yang:ietf-trans-client-service";
   prefix "clntsvc";

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

   import ietf-layer1-types {
     prefix "layer1-types";
     reference "RFC ZZZZ - A YANG Data Model for Layer 1 Types";
   }

   import ietf-yang-types {
     prefix "yang";
     reference "RFC 6991 - Common YANG Data Types";
   }

  import ietf-trans-client-svc-types {
    prefix "clntsvc-types";
    reference "RFC XXXX - A YANG Data Model for
               Transport Network Client Signals";
  }

   organization
     "Internet Engineering Task Force (IETF) CCAMP WG";
   contact
     "
       ID-draft editor:
         Haomian Zheng (zhenghaomian@huawei.com);
         Aihua Guo (aihuaguo.ietf@gmail.com);
         Italo Busi (italo.busi@huawei.com);
         Anton Snitser (asnizar@cisco.com);
         Francesco Lazzeri (francesco.lazzeri@ericsson.com);
     ";

   description
     "This module defines a YANG data model for describing
      transport network client services. The model fully conforms
      to the Network Management Datastore Architecture (NMDA).

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

```
   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).
   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

revision 2021-01-11 {
  description
    "version -04 as a WG document";
  reference
    "draft-ietf-ccamp-client-signal-yang";
}

/*
 * Groupings
 */
grouping client-svc-access-parameters {
  description
    "Transport network client signals access parameters";

  leaf access-node-id {
    type te-types:te-node-id;
    description
      "The identifier of the access node in the underlying
       transport network topology.";
  }

  leaf access-ltp-id {
    type te-types:te-tp-id;
    description
      "The TE link termination point identifier, used together with
       access-node-id to identify the access LTP.";
  }

  leaf client-signal {
    type identityref {
      base layer1-types:client-signal;
    }
    description
      "Identify the client signal type associated with this port";
  }


}

grouping client-svc-tunnel-parameters {
  description
```

```
    "Transport network client signals tunnel parameters";

  leaf tunnel-name {
    type string;
    description
      "TE tunnel instance name.";
  }
}

grouping  client-svc-instance-config {
  description
    "Configuration parameters for client services.";
  leaf client-svc-name {
    type string;
    description
      "Identifier of the p2p transport network client signals.";
  }

  leaf client-svc-title {
    type string;
    description
      "Name of the p2p transport network client signals.";
  }

  leaf client-svc-descr {
    type string;
    description
      "Description of the transport network client signals.";
  }

  leaf client-svc-customer {
    type string;
    description
      "Customer of the transport network client signals.";
  }

  container resilience {
    description "Place holder for resilience functionalities";
  }

  uses te-types:te-topology-identifier;

  leaf admin-status {
    type identityref {
      base te-types:tunnel-admin-state-type;
    }
    default te-types:tunnel-admin-state-up;
    description "Client signals administrative state.";
  }
```

```
  container src-access-ports {
    description
      "Source access port of a client signal.";
    uses client-svc-access-parameters;
  }

  container dst-access-ports {
    description
      "Destination access port of a client signal.";
    uses client-svc-access-parameters;
  }

  leaf direction {
    type identityref {
      base clntsvc-types:direction;
    }
    description "Uni-dir or Bi-dir for the client signal.";
  }

  list svc-tunnels {
    key tunnel-name;
    description
      "List of the TE Tunnels supporting the client signal.";
    uses client-svc-tunnel-parameters;
  }
}

grouping  client-svc-instance-state {
  description
    "State parameters for client services.";
  leaf operational-state {
    type identityref {
          base te-types:tunnel-state-type;
    }
    config false;
    description "Client signal operational state.";
  }
  leaf provisioning-state {
    type identityref {
          base te-types:lsp-state-type;
    }
    config false;
    description "Client signal provisioning state.";
  }
  leaf creation-time {
    type yang:date-and-time;
    config false;
    description "The time of the client signal be created.";
  }
```

```
    leaf last-updated-time {
      type yang:date-and-time;
      config false;
      description "The time of the client signal's latest update.";
    }
    leaf created-by {
      type string;
      config false;
      description
        "The client signal is created by whom,
         can be a system or staff ID.";
    }
    leaf last-updated-by {
      type string;
      config false;
      description
        "The client signal is last updated by whom,
         can be a system or staff ID.";
    }
    leaf owned-by {
      type string;
      config false;
      description
        "The client signal is owned by whom,
         can be a system ID.";
    }
  }

  /*
   * Data nodes
   */

  container client-svc {
    description
      "Transport client services.";

    list client-svc-instances {
      key client-svc-name;
      description
        "The list of p2p transport client service instances";

          uses client-svc-instance-config;
          uses client-svc-instance-state;
    }
  }
}

<CODE ENDS>
```

5.4. Other Client Signal Types YANG Code

This module defines the types for other client signal types.

```
<CODE BEGINS> file "ietf-trans-client-svc-types@2019-11-03.yang"


module ietf-trans-client-svc-types {
  namespace "urn:ietf:params:xml:ns:yang:ietf-trans-client-svc-types";
  prefix "clntsvc-types";

  organization
     "Internet Engineering Task Force (IETF) CCAMP WG";
   contact
     "
       ID-draft editor:
         Haomian Zheng (zhenghaomian@huawei.com);
         Aihua Guo (aihuaguo.ietf@gmail.com);
         Italo Busi (italo.busi@huawei.com);
         Anton Snitser (asnizar@cisco.com);
         Francesco Lazzeri (francesco.lazzeri@ericsson.com);
     ";

   description
     "This module defines a YANG data model for describing
      transport network client types. The model fully conforms
      to the Network Management Datastore Architecture (NMDA).

      Copyright (c) 2019 IETF Trust and the persons
      identified as authors of the code.  All rights reserved.

      Redistribution and use in source and binary forms, with or
      without modification, is permitted pursuant to, and subject
      to the license terms contained in, the Simplified BSD License
      set forth in Section 4.c of the IETF Trust's Legal Provisions
      Relating to IETF Documents
      (https://trustee.ietf.org/license-info).
      This version of this YANG module is part of RFC XXXX; see
      the RFC itself for full legal notices.";

   revision 2019-11-03 {
     description
       "version -01 as a WG document";
     reference
       "draft-ietf-ccamp-client-signal-yang";
   }

  identity direction {
    description
      "Direction information of Client Signal.";
  }

  identity bidirectional {
    base direction;
```

```
    description
      "Client Signal is bi-directional.";
  }

  identity unidirectional {
    base direction;
    description
      "Client Signal is uni-directional.";
  }

}

<CODE ENDS>
```

6. Implementation Status

[Note to the RFC Editor - remove this section before publication, as well as remove the reference to RFC [RFC7942].]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

6.1. Usage of the ETH Service YANG Model on ONAP

The implementation of the CCVPN (Cross Domain and Cross Layer VPN) use-case on ONAP follows the ACTN [RFC8453] architecture. In the design of CCVPN, ONAP presumes the responsibility of the MDSC, and third party network domain controllers the PNCs. Consequently, the ETH Service YANG model is used as the MPI between ONAP and the domain controllers.

*Organization: China Mobile, Huawei Technologies, etc.

*Implementation: ONAP CCVPN uses the ETH Service YANG model as the ACTN MPI

*Description: ONAP CCVPN realizes the E-LINE and E-TREE service on a multi-domain network. Both of the services are modeled on ONAP by the ETH Service YANG model, and the model instances (e.g., JSON objects) are sent between ONAP and the domain controllers. Refer to the following CCVPN wiki for more information: https://wiki.onap.org/display/DW/CCVPN%28Cross+Domain+and+Cross+Layer+VPN%29+USE+CASE

*Maturity Level: Prototype

*Coverage: Partial

*Contact: henry.yu1@huawei.com

7. IANA Considerations

It is proposed that IANA should assign new URIs from the "IETF XML Registry" [RFC3688] as follows:

URI: urn:ietf:params:xml:ns:yang:ietf-eth-tran-service
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-trans-client-service
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-eth-tran-types
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-trans-client-svc-types
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

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

```
name:         ietf-eth-tran-service
namespace:    urn:ietf:params:xml:ns:yang:ietf-eth-tran-service
prefix:       ethtsvc
reference:    RFC XXXX: A YANG Data Model for Transport
                        Network Client Signals

name:         ietf-eth-tran-types
namespace:    urn:ietf:params:xml:ns:yang:ietf-eth-tran-types
prefix:       etht-types
reference:    RFC XXXX: A YANG Data Model for Transport
                        Network Client Signals

name:         ietf-trans-client-service
namespace:    urn:ietf:params:xml:ns:yang:ietf-trans-client-service
prefix:       clntsvc
reference:    RFC XXXX: A YANG Data Model for Transport
                        Network Client Signals

name:         ietf-trans-client-svc-types
namespace:    urn:ietf:params:xml:ns:yang:ietf-trans-client-svc-types
prefix:       clntsvc-types
reference:    RFC XXXX: A YANG Data Model for Transport
                        Network Client Signals
```

8. Manageability Considerations

TBD.

9. Security Considerations

The data following the model defined in this document is exchanged via, for example, the interface between an orchestrator and a network domain controller.

The YANG module defined in this document can be accessed via the RESTCONF protocol defined in [RFC8040], or maybe via the NETCONF protocol [RFC6241].

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., POST) to these data nodes without proper protection can have a negative effect on network operations.

10. Acknowledgements

We would like to thank Igor Bryskin and Daniel King for their comments and discussions.

11. Contributors

Yunbin Xu CAICT Email: xuyunbin@caict.ac.cn

Yang Zhao China Mobile Email: zhaoyangyjy@chinamobile.com

Xufeng Liu Volta Networks Email: xufeng.liu.ietf@gmail.com

Giuseppe Fioccola Huawei Technologies Email: giuseppe.fioccola@huawei.com

Yanlei Zheng China Unicom Email: zhengyanlei@chinaunicom.cn

Zhe Liu Huawei Technologies, Email: liuzhe123@huawei.com

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

Yingxi Yao Shanghai Bell, yingxi.yao@nokia-sbell.com

Henry Yu Huawei Technologies, henry.yu1@huawei.com

12. References

12.1. Normative References

[I-D.ietf-ccamp-l1csm-yang] Lee, Y., Lee, K., Zheng, H., de Dios, O. G., and D. Ceccarelli, "A YANG Data Model for L1 Connectivity Service Model (L1CSM)", Work in Progress, Internet-Draft, draft-ietf-ccamp-l1csm-yang-23, 3 April 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ccamp-l1csm-yang-23>.

[I-D.ietf-ccamp-layer1-types] Zheng, H. and I. Busi, "A YANG Data Model for Layer 1 Types", Work in Progress, Internet-Draft, draft-ietf-ccamp-layer1-types-15, 23 November 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ccamp-layer1-types-15>.

[I-D.ietf-ccamp-otn-topo-yang] Zheng, H., Busi, I., Liu, X., Belotti, S., and O. G. de Dios, "A YANG Data Model for Optical Transport Network Topology", Work in Progress, Internet-Draft, draft-ietf-ccamp-otn-topo-yang-16, 23 November 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ccamp-otn-topo-yang-16>.

[I-D.ietf-ccamp-otn-tunnel-model] Zheng, H., Busi, I., Belotti, S., Lopez, V., and Y. Xu, "OTN Tunnel YANG Model", Work in Progress, Internet-Draft, draft-ietf-ccamp-otn-tunnel-model-18, 3 April 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ccamp-otn-tunnel-model-18>.

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

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

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

[RFC7139] Zhang, F., Ed., Zhang, G., Belotti, S., Ceccarelli, D., and K. Pithewan, "GMPLS Signaling Extensions for Control of Evolving G.709 Optical Transport Networks", RFC 7139, DOI 10.17487/RFC7139, March 2014, <https://www.rfc-editor.org/info/rfc7139>.

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

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

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

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

12.2. Informative References

[IEEE802.1ad] IEEE, 802.1ad., "IEEE 802.1ad - Provider Bridges.", IEEE 802.1ad , May 2006.

[IEEE802.1q] IEEE, 802.1q., "IEEE 802.1q - Virtual Bridged Local Area Networks", IEEE 802.1q , June 2005.

[MEF10] MEF, 10., "Ethernet Services Attributes Phase 1", MEF10 , November 2004.

[RFC2697] Heinanen, J. and R. Guerin, "A Single Rate Three Color Marker", RFC 2697, DOI 10.17487/RFC2697, September 1999, <https://www.rfc-editor.org/info/rfc2697>.

[RFC2698] Heinanen, J. and R. Guerin, "A Two Rate Three Color Marker", RFC 2698, DOI 10.17487/RFC2698, September 1999, <https://www.rfc-editor.org/info/rfc2698>.

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

[RFC4115] Aboul-Magd, O. and S. Rabie, "A Differentiated Service Two-Rate, Three-Color Marker with Efficient Handling of in-Profile Traffic", RFC 4115, DOI 10.17487/RFC4115, July 2005, <https://www.rfc-editor.org/info/rfc4115>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

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

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

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

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

[RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for Abstraction and Control of TE Networks (ACTN)", RFC 8453,

DOI 10.17487/RFC8453, August 2018, <https://www.rfc-editor.org/info/rfc8453>.

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

Authors' Addresses

Haomian Zheng
Huawei Technologies
H1 XiliuBeipo Village, Songshan Lake
Dongguan
Guangdong,
China

Email: zhenghaomian@huawei.com

Aihua Guo
Futurewei

Email: aihuaguo.ietf@gmail.com

Italo Busi
Huawei Technologies

Email: Italo.Busi@huawei.com

Anton Snitser
Cisco

Email: asnizar@cisco.com

Francesco Lazzeri
Ericsson

Email: francesco.lazzeri@ericsson.com