A YANG Data Model for Transport Network Client Signals
draft-zheng-ccamp-client-signal-yang-03

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 Traffic-engineered models and OTN models, however, the access to the network has not been described. These information is useful to both client and provider.

This draft describe how the client signals are carried over transport network and defined corresponding YANG data model which is 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 April 25, 2019.
1. Introduction

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 there has been topology and tunnel model defined for transport network, such as [I-D.ietf-ccamp-otn-topo-yang] and [I-D.ietf-ccamp-otn-tunnel-model], which has described the network model between PEs. However, there is a missing piece for the mapping...
between the PE and the CE, which 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 controller via a REST 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 model defined in this document is independent of control plane protocols and captures topology related information. Furthermore, it is not a stand-alone model, but augmenting from the TE topology YANG model defined in [I-D.ietf-teas-yang-te-topo].

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 [I-D.ietf-netmod-yang-tree-diagrams]. 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

The transport network is usually a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic opaquely across the server-layer network resources. A transport network may be constructed from equipments utilizing any of a number of different transport technologies such as the evolving optical transport infrastructure (SONET/SDH and OTN) or packet transport as epitomized by the MPLS Transport Profile (MPLS-TP).

In the example of OTN as the 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 Ethernet, ODU, STM-n and so on. In this approach the OTN devices do not is not aware of the client signal type, and this information is only necessary among the controllers. Once 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 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 in this draft are applicable for both of the two above categories.

It is also worth noting that some client signal can be carried over multiple types of transport 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.

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? ethht-types:bandwidth-profile-type
        +--rw CIR? uint64
        +--rw CBS? uint64
        +--rw EIR? uint64
        +--rw EBS? uint64
        +--rw color-aware? boolean
        +--rw coupling-flag? boolean
4.2. YANG Tree for other Transport Network Client Signal Model

```yang
module: ietf-trans-client-service
  +--rw client-svc
    +--rw client-svc-instances* [client-svc-name]
    |  +--rw client-svc-name           string
    |  +--rw client-svc-descr?         string
    |  +--rw te-topology-identifier
    |     |  +--rw provider-id?   te-types:te-global-id
    |     |  +--rw client-id?     te-types:te-global-id
    |     |  +--rw topology-id?   te-types:te-topology-id
    |  +--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 svc-tunnels* [tunnel-name]
    |  +--rw tunnel-name    string
```

5. YANG Code for Transport Network Client Signal

5.1. The ETH Service YANG Code

```yang
<CODE BEGINS> file "ietf-eth-tran-service@2018-10-18.yang"

module ietf-eth-tran-service {

  namespace "urn:ietf:params:xml:ns:yang:ietf-eth-tran-service";
  prefix "ethtsvc";

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-te-types {
    prefix "te-types";
  }

  ...
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 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 bandwidth 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 "string";
        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 bandwidth 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 bandwidth profile used in the ingress direction."
    uses named-or-value-bandwidth-profile;
  }
  container egress-bandwidth-profile {
    description
    "The bandwidth profile used in the egress direction."
    uses named-or-value-bandwidth-profile;
  }
}
}

grouping etht-svc-access-parameters {
  description
  "ETH transport services access parameters";
  leaf access-node-id {
    type te-types:te-node-id;
    description
    "The identifier of the access node in
    the ETH transport 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 service-classification-type {
    type identityref {
      base etht-types: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;
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 etht-svc-tunnel-parameters {
  /*
   * 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 etht-svc-tunnel-parameters {
description
  "ETH transport services tunnel parameters";

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

choice svc-multiplexing-tag {
  description
    "Service multiplexing is optional and flexible.";

  case other {
    /*
     * placeholder to support proprietary multiplexing
     * (for further discussion)
    */
  }

  case none {
    /* no additional information is needed */
  }

  case vlan-tag {
    /*
     * No additional information is needed
     * The C-Tag or S-Tag used for service multiplexing is defined
     * by the VLAN classification and operations configured in the
     * eth-tsvc-access-parameters grouping
    */
  }

  case pw {
    /* to be completed (for further discussion) */
  }
}

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 "Configuration 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
  "Configuration parameters for Ethernet services."
  leaf etht-svc-name {
    type string;
    description
    "Name of the ETH transport service."
  }
  leaf etht-svc-id {
    type yang:uuid;
    description
    "Universally Unique IDentifier (UUID) of the ETH transport service."
  }
  leaf etht-svc-descr {
    type string;
    description
    "Description of the ETH transport service."
  }
  leaf etht-svc-customer {
    type string;
    description
    "Customer of the ETH transport service."
  }
  leaf etht-svc-type {
    type etht-types:service-type;
    description
    "Type of ETH transport service (p2p, mp2mp or rmp)."
    /* Add default as p2p */
  }
  leaf etht-svc-lifecycle {
    type etht-types:lifecycle-status;
    description
    "Lifecycle state of ETH transport service."
    /* Add default as installed */
  }
}
uses te-types:te-topology-identifier;

list etht-svc-access-ports {
  key access-port-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 port instances.";
  leaf access-port-id {
    type uint16;
    description
      "ID of the service access port instance";
  }
  uses etht-svc-access-parameters;
}
list etht-svc-tunnels {
  key tunnel-name;
  description
    "List of the TE Tunnels supporting the ETH transport service.";
  uses etht-svc-tunnel-parameters;
}
container pm-config {
  description
    "ETH service performance monitoring";
  leaf pm-enable {
    type boolean;
    description
      "Boolean value indicating whether PM is enabled.";
  }
  uses etht-svc-pm-threshold_config;
}
leaf admin-status {
  type identityref {
    base te-types:tunnel-state-type;
  }
  default te-types:tunnel-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.";
  }
  uses etht-svc-pm-stats;
}

/*
 * Data nodes
 */

container etht-svc {
  description "ETH transport 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 {

```
5.2. YANG Code for ETH transport type

<CODE BEGINS> file "ietf-eth-tran-types@2018-10-18.yang"
module ietf-eth-tran-types {

    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);

<CODE ENDS>
description
  "This module defines the ETH transport types.";

revision 2018-10-18 {
  description
    "Initial Revision";
  reference
    "draft-zheng-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;
"Committed Information Rate in Kbps"

} leaf CBS {
    type uint64;
    description
        "Committed Burst Size 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.";
}
5.3. Other Transport Network client signal YANG Code

<CODE BEGINS> file "ietf-trans-client-service@2018-10-19.yang"
module ietf-trans-client-service { 
   /* TODO: FIXME */
   //yang-version 1.1;

   prefix "clntsvc";

   import ietf-te-types { 
      prefix "te-types";
   }

   import ietf-l1-service-types { 
      prefix "ll-svc-types";
   }

   organization
      "Internet Engineering Task Force (IETF) CCAMP WG";
   contact
      "ID-draft editor:
   Aihua Guo (aihuaguo@huawei.com);
   Haomian Zheng (zhenghaomian@huawei.com);
   Italo Busi (italo.busi@huawei.com);
   Yunbin Xu (xuyunbin@ritt.cn);
   Yang Zhao (zhaoyangjy@chinamobile.com);
   Xufeng Liu (Xufeng_Liu@jabil.com);
   Giuseppe Fioccola (giuseppe.fioccola@telecomitalia.it);
      ";

description
   "This module defines a YANG data model for describing
   simple transport client services.";

revision 2018-10-19 { 
   description
      "Initial version";
   reference
      "draft-zheng-ccamp-client-signal-yang";
}
/
 * Groupings
 */

grouping client-svc-access-parameters {
  description
   "Transport client services access parameters";

  leaf access-node-id {
    type te-types:te-node-id;
    description
     "The identifier of the access node in the underlying transport 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 l1-svc-types:protocol-type;
    }
    description
     "Identifies the client signal type associated with this port";
  }
}

grouping client-svc-tunnel-parameters {
  description
   "Transport client services tunnel parameters";

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

grouping client-svc-instance_config {
  description
   "Configuraiton parameters for client services.";

  leaf client-svc-name {
    type string;
    description
     "Name of the p2p transport client service.";
  }
}
leaf client-svc-descr {
    type string;
    description
        "Description of the transport client service.";
}
uses te-types:te-topology-identifier;
container src-access-ports {
    description
        "Source access port of a client service.";
    uses client-svc-access-parameters;
}
container dst-access-ports {
    description
        "Destination access port of a client service.";
    uses client-svc-access-parameters;
}
list svc-tunnels {
    key tunnel-name;
    description
        "List of the TE Tunnels supporting the client service.";
    uses client-svc-tunnel-parameters;
}
leaf admin-status {
    type identityref {
        base te-types:tunnel-state-type;
    }
    default te-types:tunnel-state-up;
    description "Client service administrative state.";
}
}
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 service operational state.";
    }
    leaf provisioning-state {
        type identityref {
            base te-types:lsp-state-type;
        }
        config false;
        description "Client service provisioning state.";
    }
}
6. Considerations and Open Issue

Editor Notes: This section is used to note temporary discussion/conclusion that to be fixed in the future version, and will be removed before publication. We currently categorize all the client signal types into transparent and non-transparent, with separate models. There was consensus that no common model is needed for these two categories.

7. IANA Considerations

TBD.

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 transport network controller. The security concerns mentioned in
[I-D.ietf-teas-yang-te-topo] for using ietf-te-topology.yang model also applies to this document.

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

Yanlei Zheng
China Unicom
Email: zhengyl@dimpt.com

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

12. References

12.1. Normative References

[I-D.ietf-ccamp-otn-topo-yang]

Authors’ Addresses

Haomian Zheng
Huawei Technologies
H1-1-A043S Huawei Industrial Base, Songshanhu Dongguan, Guangdong P.R.China

Email: zhenghaomian@huawei.com

Aihua Guo
Huawei Technologies

Email: aihuaguo@huawei.com

Italo Busi
Huawei Technologies

Email: Italo.Busi@huawei.com

Yunbin Xu
CAICT

Email: xuyunbin@ritt.cn

Yang Zhao
China Mobile

Email: zhaoyangyjy@chinamobile.com

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com