Basic YANG Model for Steering Client Services To Server Tunnels
draft-bryskin-teas-service-tunnel-steering-model-02

Abstract

This document describes a YANG data model for managing pools of transport tunnels and steering client services on them.

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 August 30, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect
1. Introduction

Client layer services/signals are normally mapped onto carrying them across the network transport tunnels via client/server layer adaptation relationships. Such relationships are usually modeled as multi-layer topologies, whereas tunnels set up in underlay (server) topologies support links in respective overlay (client) topologies. In this respect having a link in a client topology means that the client layer traffic could be forwarded between link termination points (LTPs) terminating the link on opposite sides by the supporting tunnel(s) provisioned in the server layer topology.

This said there are numerous use cases in which describing the client service to server tunnel bindings via the topology formalism is impractical. Below are some examples of such use cases:

- Mapping client services onto tunnels within the same network layer, for example, mapping L3 VPNs or MPLS-SR services onto IP MPLS tunnels;

- Mapping client services onto tunnels provisioned in the highest layer topology supported by the network. For example, mapping L2VPNs or E(V)PL services onto IP MPLS tunnels provisioned in an IP network;
Mapping client services to tunnels provisioned in separate network layers at the network’s access points. Consider, for example, an OTN/ODUk network that is used to carry client signals of, say, 20 different types (e.g. Ethernet, SDH, FKON, etc.) entering and exiting the network over client facing interfaces. Although it is possible to describe such a network as a 21-layer TE topology with the OTN/ODUk topology serving each of the 20 client layer topologies [I-D.ietf-teas-yang-te-topo], such a description would be verbose, cumbersome, difficult to expand to accommodate additional client signals and unnecessary, because the client layer topologies would have zero switching flexibility inside the network (i.e. contain only unrelated links connecting access points across respective layer networks), and all what is required to know from the point of view of a management application is what ODUk tunnels are established or required, which client signals the tunnels could carry and at which network border nodes and how the client signals could be bound (i.e. adopted) to the tunnels.

It is worth noting that such non-topological client-service-to-server-tunnel mapping almost always happens on network border nodes. However, there are also important use cases where such a mapping is required in the middle of the network. One such use case is controlling on IP/MPLS FRR PLRs which LSPs are mapped onto which backup tunnels.

It is important to bear in mind that service2tunnel mappings could be very complex: large number of instances of services of the same or different types (possibly governed by different models) could be mapped on the same set of tunnels, with the latter being set in different network layers and of either TE or non-TE nature, P2P or P2MP or MP2MP type. Furthermore, the mappings could be hierarchical: tunnels carrying services could be clients of other tunnels.

Despite of the differences of transport tunnels and of services they carry the service2tunnel mappings could be modeled in a simple uniform way. Access to a data store of such mappings could be beneficial to network management applications. It would be possible, for example, to discover which services depend on which tunnels, which services will be affected if a given tunnel goes out of service, how many more services could be placed onto a given TE tunnel without the latter violating its TE commitments (such as bandwidth and delay). It would be also possible to demand in a single request moving numerous (ranges of) service instances from one set of tunnels to another.

This document defines a YANG data model for facilitating said service2tunnel mappings.
The YANG model in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

1.1. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119].

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

- augment
- data model
- data node

1.2. Tree Diagrams

A simplified graphical representation of the data model is presented in this document, by using the tree format defined in [RFC8340].

1.3. Prefixes in Data Node Names

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

| Prefix   | YANG module     | Reference               |
|----------+-----------------+-------------------------|
| inet     | ietf-inet-types | [RFC6991]               |
| te-types | ietf-te-types   | [I-D.ietf-teas-yang-te] |

Table 1: Prefixes and Corresponding YANG Modules

2. Explicit vs. Implicit Service2tunnel Mapping. Steering Services to Transport Tunnel Pools

There are use cases in which client services require hard separation of the transport carrying them from the transport carrying other services. However, environment in which the services may share the same transport tunnels is far more common. For this reason the model
defined in this document suggests replacing (or at least augmenting) the explicit service2tunnel mapping configuration (in which the tunnels are referred to by their IDs/names) with an implicit mapping. Specifically, the model introduces the notion of tunnel pool. A tunnel pool could be referred to by its network unique color and requires a service2tunnel mapping configuration to specify the tunnel pool color(s) instead of tunnel IDs/names. The model governs tunnel pool data store independently from the services steered on the tunnels. It is assumed (although not required) that the tunnels - constituents/components of a tunnel pool - are of the same type, provisioned using a common template. Importantly they could be dynamically added to/removed from the pool without necessitating service2tunnel mapping re-configuration. Such a service to tunnel pool steering approach has the following advantages:

- Scalability and efficiency: pool component bandwidth utilization could be monitored, tunnels could be added to/removed from the pool if/when detected that current component bandwidth utilization has crossed certain thresholds. This allows for a very efficient network resource utilization and obviates the network management application from a very difficult task of service to tunnel mapping planning;

- Automation and elasticity: pool component attributes could be modified - bandwidth auto-adjusted, protection added, delay constrained, etc.. The tunnels could be completely or partially replaced with tunnels of different types (e.g. TE vs. non-TE, P2P vs. P2MP, etc.) or even provisioned in different network layers (OTN/ODuk tunnels replacing IP TE tunnels). Importantly, all such modifications do not require service2tunnel mapping re-configurations as long as the modified or new tunnels remain within the same tunnel pool(s);

- Transparency: new service sites supported by additional PEs could be added without service2tunnel mapping re-configuration.

3. The purpose of the model

The model is targeted to facilitate for network management applications, such as service orchestrators, the control of pools of transport tunnels and steering onto them client services independently of network technology/layer specifics of both the services and the tunnels. The model could be applied to/implemented on physical devices, such as IP routers, as well as on abstract topology nodes. Furthermore, the model could be supported by a network (domain) controller, such as ACTN PNC, to act as a proxy server on behalf of any network element/node (physical or abstract) under its control.
4. Model Design

The data store described/governed by the model is comprised of a single top level list - TunnelPools. A TunnelPool, list element, is a container describing a set of transport tunnels (presumably with similar characteristics) identified by a network unique ID (color).

The TunnelPool container has the following fields:

- Color [uint32 list key];
- Tunnels list;
- Services list.

The Tunnels list describes the pool constituents - active transport tunnels. The list members - Tunnel containers - include the following information:

- tunnel type [e.g. P2P-TE, P2MP-TE, SR-TE, SR P2P, LDP P2P, LDP MP2MP, GRE, PBB, etc]
- tunnel type specific tunnel ID [provided that a data store of the tunnel type, e.g. TE tunnels, is supported, the tunnelID allows for the management application to look up the tunnel in question to obtain detailed information about the tunnel];
- topology ID [identifies the topology over which the tunnel’s connection paths are defined];
- tunnel encapsulation [e.g. MPLS label stack, Ethernet STAGs, GRE header, PBB header, etc].

The Services list describes services currently steered on the tunnel pool. The list members - Service containers - have the following attributes:

- service type [e.g. fixed/transparent, L3VPN, L2VPN, EVPN, ELINE, EPL, EVPL, L1VPN, ACTN VN, etc.];
- service type specific service ID [provided that a data store of the service type, e.g. L2VPN, is supported, the service ID allows for the management application to look up the service in question to obtain detailed information about the service];
- client ports (source/destination node LTPs over which the service enters/ exits the node/network, relevant only for fixed/transparent services);
5. Tree Structure

module: ietf-tunnel-steering
  ++-rw tunnel-pools
  ++-rw tunnel-pool* [color]
    ++-rw color          uint32
    ++-rw description?   string
  ++-rw service* [service-type id]
    ++-rw service-type     identityref
    ++-rw id               string
    ++-rw encapsulation    binary
    |  ++-rw type?    identityref
    |  ++-rw value?   binary
  ++-rw access-point* [node-address link-termination-point]
    ++-rw node-address              inet:ip-address
    ++-rw link-termination-point     string
  ++-rw direction?                enumeration
  ++-rw tunnel* [provider-id client-id topology-id tunnel-type source destination tunnel-id]
    ++-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 tunnel-type      identityref
    ++-rw source           inet:ip-address
    ++-rw destination      inet:ip-address
    ++-rw tunnel-id        binary
    ++-rw encapsulation    binary
    |  ++-rw type?    identityref
    |  ++-rw value?   binary

6. YANG Modules

<CODE BEGINS> file "ietf-tunnel-steering@2019-02-15.yang"
module ietf-tunnel-steering {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-tunnel-steering";
  prefix "tnl-steer";
</CODE BEGINS>
import ietf-inet-types {
  prefix inet;
}

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

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

contact
  "WG Web: <http://tools.ietf.org/wg/teas/>
  WG List: <mailto:teas@ietf.org>
  Editors: Igor Bryskin
    <mailto:Igor.Bryskin@huawei.com>
  Editor: Vishnu Pavan Beeram
    <mailto:vbeeram@juniper.net>
  Editor: Tarek Saad
    <mailto:tsaad@cisco.com>
  Editor: Xufeng Liu
    <mailto:xufeng.liu.ietf@gmail.com>
  Editor: Young Lee
    <mailto:leeyoung@huawei.com> ";

description
  "This data model is for steering client service to server tunnels."

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

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

revision 2019-02-15 {
  description "Initial revision";
  reference "TBD";
typedefs

identities

identity service-type {
    description "Base identity for client service type.";
}

identity service-type-l3vpn {
    base service-type;
    description "L3VPN service.";
}

identity service-type-l2vpn {
    base service-type;
    description "L2VPN service.";
}

identity service-type-evpn {
    base service-type;
    description "EVPN service.";
}

identity service-type-eline {
    base service-type;
    description "ELINE service.";
}

identity service-type-epl {
    base service-type;
    description "EPL service.";
}

identity service-type-evpl {
    base service-type;
    description "EVPL service.";
}

identity service-type-l1vpn {
    base service-type;
    description "L1VPN service.";
}

identity service-type-actn-vn {
base service-type;
description
"ACTN VN service.";
}

identity service-type-transparent {
  base service-type;
  description
  "Transparent LAN service.";
}

identity tunnel-type {
  description "Base identity for tunnel type.";
}

identity tunnel-type-te-p2p {
  base tunnel-type;
  description
  "TE point-to-point tunnel type.";
}

identity tunnel-type-te-p2mp {
  base tunnel-type;
  description
  "TE point-to-multipoint tunnel type.";
  reference "RFC4875";
}

identity tunnel-type-te-sr {
  base tunnel-type;
  description
  "Segment Rouging TE tunnel type.";
}

identity tunnel-type-sr {
  base tunnel-type;
  description
  "Segment Rouging tunnel type.";
}

identity tunnel-type-ldp-p2p {
  base tunnel-type;
  description
  "LDP point-to-point tunnel type.";
}

identity tunnel-type-ldp-mp2mp {
  base tunnel-type;
  description
  "Multicast LDP multipoint-to-multipoint tunnel type.";
}

identity tunnel-type-gre {
  base tunnel-type;
  description
  "GRE tunnel type.";
}
identity tunnel-type-pbb {
    base tunnel-type;
    description
        "PBB tunnel type.";
}

identity service-encapsulation-type {
    description "Base identity for tunnel encapsulation.";
}

identity service-encapsulation-type-mpls-label {
    base service-encapsulation-type;
    description
        "Encapsulated by MPLS label stack, as an inner label to
        identify the customer service.";
}

identity service-encapsulation-type-ethernet-c-tag {
    base service-encapsulation-type;
    description
        "Encapsulated by Ethernet C-TAG, to identify the customer
        service.";
}

identity tunnel-encapsulation-type {
    description "Base identity for tunnel encapsulation.";
}

identity tunnel-encapsulation-type-mpls-label {
    base tunnel-encapsulation-type;
    description
        "Encapsulated by MPLS label stack, as an outer label to
        be pushed into the tunnel.";
}

identity tunnel-encapsulation-type-ethernet-s-tag {
    base tunnel-encapsulation-type;
    description
        "Encapsulated by Ethernet S-TAG.";
}

identity tunnel-encapsulation-type-pbb {
    base tunnel-encapsulation-type;
    description
        "Encapsulated by PBB header.";
}

identity tunnel-encapsulation-type-gre {
    base tunnel-encapsulation-type;
    description
        "Encapsulated by GRE header.";
}
/ * Groupings /

="/*
 * Configuration data and operational state data nodes
 */
container tunnel-pools {
  description
  "A list of mappings that steer client services to transport tunnel pools. The tunnel pools are managed independently from the services steered on them.";

list tunnel-pool {
  key "color";
  description
  "A set of transport tunnels (presumably with similar characteristics) identified by a network unique ID, named 'color'.";
  leaf color {
    type uint32;
    description
    "Unique ID of a tunnel pool.";
  }
  leaf description {
    type string;
    description
    "Client provided description of the tunnel pool.";
  }
list service {
  key "service-type id";
  description
  "A list of client services that are steered on this tunnel pool.";
  leaf service-type {
    type identityref {
      base service-type;
    }
    description
    "Service type required by the client.";
  }
  leaf id {
    type string;
    description
    "Unique ID of a client service for the specified service type.";
  }
  container encapsulation {
}
description
"The encapsulation information used to identify the
customer service for multiplexing over shared tunnels.";
leaf type {
    type identityref {
        base service-encapsulation-type;
    }
    description
    "The encapsulation type used to identify the customer
    service for multiplexing over shared tunnels.";
}
leaf value {
    type binary;
    description
    "The encapsulation value pushed to the tunnel to
    identify this service.
    If not specified, the system decides what
    value to be used for multiplexing.";
}
list access-point {
    key "node-address link-termination-point";
    description
    "A list of client ports (Link Termination Points) for the
    service to enter or exist.";
    leaf node-address {
        type inet:ip-address;
        description
        "Node over which the service enters or exists.";
    }
    leaf link-termination-point {
        type string;
        description
        "Client port (Link Termination Point) over which the
        service enters or exits.";
    }
    leaf direction {
        type enumeration {
            enum "in" {
                description "The service enters to the network.";
            }
            enum "out" {
                description "The service exists from the network.";
            }
            enum "in-out" {
                description
                "The service enters to and exists from the
                network.";
            }
        }
    }
}
list tunnel {
  key "provider-id client-id topology-id tunnel-type source " + "destination tunnel-id";
  description "A list of tunnels in the tunnel pool.";
  leaf provider-id {
    type te-types:te-global-id;
    description "An identifier to uniquely identify a provider.";
  }
  leaf client-id {
    type te-types:te-global-id;
    description "An identifier to uniquely identify a client.";
  }
  leaf topology-id {
    type te-types:te-topology-id;
    description "It is presumed that a datastore will contain many topologies. To distinguish between topologies it is vital to have UNIQUE topology identifiers.";
  }
  leaf tunnel-type {
    type identityref {
      base tunnel-type;
    }
    description "Tunnel type based on constructing technologies and multipoint types, including P2P-TE, P2MP-TE, SR-TE, SR P2P, LDP P2P, LDP MP2MP, GRE, PBB, etc";
  }
  leaf source {
    type inet:ip-address;
    description "For a p2p or p2mp tunnel, this is the source address; for a mp2mp tunnel, this is the root address.";
    reference "RFC3209, RFC4875, RFC6388, RFC7582.";
  }
  leaf destination {

type inet:ip-address;
description "For a p2p tunnel, this is the tunnel endpoint address extracted from SESSION object;
for a p2mp tunnel, this identifies the destination group, or p2mp-id;
for a mp2mp tunnel identified by root and opaque-value, this value is set to '0.0.0.0'."
reference "RFC3209, RFC4875, RFC6388, RFC7582.";
}
leaf tunnel-id {
  type binary;
  description "For a p2p or p2mp tunnel, this is the tunnel identifier used in the SESSION that remains constant over the life of the tunnel;
for a mp2mp tunnel, this is the opaque-value in the FEC element.";
  reference "RFC3209, RFC4875, RFC6388, RFC7582.";
}
container encapsulation {
  description "The encapsulation information used by the tunnel.";
  leaf type {
    type identityref {
      base service-encapsulation-type;
    }
    description "The encapsulation type used by the tunnel.";
  }
  leaf value {
    type binary;
    description "The encapsulation value pushed to the tunnel data to identify the traffic in this tunnel.
If not specified, the system decides what value to be used.";
  }
}
7. IANA Considerations

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

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

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

This document registers the following YANG modules in the YANG Module Names registry [RFC7950]:

```
name:         ietf-tunnel-steering
prefix:       tnl-steer
reference:    RFC XXXX
```

8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that 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., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

```
/tunnel-pools/tunnel-pool
```
This subtree specifies a list of tunnel pools. Modifying the configurations cause interruption to related services and tunnels.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/tunnel-pools/tunnel-pool
Unauthorized access to this subtree can disclose the information of related services and tunnels.

9. References

9.1. Normative References


9.2. Informative References


Authors' Addresses

Igor Bryskin
Huawei Technologies
EMail: Igor.Bryskin@huawei.com

Vishnu Pavan Beeram
Juniper Networks
EMail: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc
EMail: tsaad@cisco.com
Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on September 11, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in
Abstract

Multi-protocol Label Switching - Transport Profile (MPLS-TP) is a profile of the MPLS protocol that is used in packet switched transport networks and operated in a similar manner to other existing transport technologies (e.g., OTN), as described in RFC5921. This document specifies YANG models for MPLS-TP, which have not been covered by existing models so far. The gap analysis with current relevant traffic-engineering (TE) and MPLS models is also included.

Table of Contents

1. Introduction..............................................................................2
2. Considerations on the Augmentation.................................3
   2.1. Modules Relationship.............................................3
   2.2. Prefixs in Model Names........................................4
3. Gap Analysis for MPLS-TP topology...............................4
   3.1. TE Bandwidth Augmentations................................5
   3.2. TE Label Augmentations......................................6
4. Gap Analysis for MPLS-TP Tunnel Configuration...............6
   4.1. TE Bandwidth Augmentation..................................7
   4.2. TE Label Augmentation......................................7
5. Related YANG Code..........................................................8
   5.1. YANG Code for MPLS-TP Topology Augmentation...........8
   5.2. YANG Code for MPLS-TP Tunnel Augmentation.............11
   5.3. MPLS-TP Specific YANG Types.............................13
6. Open Issues............................................................................15
7. Security..................................................................................16
8. Acknowledgements..............................................................16
9. References...............................................................................16
   9.1. Normative References..........................................16
   9.2. Informative References.......................................16
Authors’ Addresses.................................................................17

1. Introduction

Multi-protocol Label Switching - Transport Profile (MPLS-TP) is a packet switching technology intended operated in a similar manner to other existing transport technologies (e.g., OTN), as described in [RFC5921], which includes Traffic Engineering (TE) features.
Generic TE models, including the TE topology and tunnel, have been defined in [TE-Topology] and [TE-Tunnel] using the YANG data modeling language and are applicable to any TE technologies including MPLS-TE and OTN and therefore also to MPLS-TP.

The YANG models for MPLS with TE features (MPLS-TE), are provided in [TE-MPLS] as a technology-specific augmentations of the generic TE models. However, technology-specific augmentations for TE label, and TE bandwidth of TE Topology and Tunnel models, have not been covered yet.

This document defines YANG data models for MPLS-TP topologies and tunnels, providing the minimum set of attributes that are required and not yet available in existing TE and MPLS YANG models. See section 3 and 4 for more detailed gap analysis.

The proposed MPLS-TP YANG models can be used as an input to enhance the current MPLS-TE YANG models.

2. Considerations on the Augmentation

2.1. Modules Relationship

In this draft two models are proposed: one MPLS-TP technology-specific topology model that augments the ietf-te-topology YANG module, defined in [TE-Topology], and another MPLS-TP technology-specific tunnel model that augments the ietf-te YANG module, defined in [TE-Tunnel].

The following common fundamental models are imported:

- o ietf-routing-types defined in [RFC8294]

```
+----------------+       +------------+    o: augment
| ietf-te-topology |       |  ietf-te   |
+----------------+       +------------+
                                  o
                                  |
                                  |
                                  |
                                  |
                                  |                  +------------------------+
                                  |                  | ietf-te-mpls-tp-topo |
                                  |                  +------------------------+
                                  |
                                  |
                                  |
                                  |                  +------------------------+
                                  |                  | ietf-te-mpls-tp-tunnel |
                                  |                  +------------------------+

Figure 1: Relationship of MPLS-TP topology and tunnel module with TE generic TE topology and tunnel YANG modules
```
2.2. Prefix 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, as shown in Table 1.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>rt-types</td>
<td>ietf-routing-types</td>
<td>[RFC8294]</td>
</tr>
<tr>
<td>tet</td>
<td>ietf-te-topology</td>
<td>[TE-Topology]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[TE-Tunnel]</td>
</tr>
<tr>
<td>te-mpls</td>
<td>ietf-te-mpls</td>
<td>[TE-MPLS]</td>
</tr>
<tr>
<td>mpls-tp-topo</td>
<td>ietf-mpls-tp-topo</td>
<td>This document</td>
</tr>
<tr>
<td>mpls-tp-tunnel</td>
<td>ietf-mpls-tp-tunnel</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

3. Gap Analysis for MPLS-TP topology

There are no YANG models that provide MPLS-TE technology-specific augmentations of the generic TE Topology model defined in [TE-Topology].

This section analyses the minimum set of attributes that are required to be specified in an MPLS-TP technology-specific augmentation.

Additional attributes that may be required to support a broader set of MPLS-TP and/or MPLS-TE functions are for further study.

Given the guidance for augmentation in [TE-Topology], the following technology-specific augmentations need to be provided:

- A network-type to indicate that the TE topology is an MPLS-TP Topology, as follow:

  augment /nw:networks/nw:network/nw:network-types/tet-te-topology:
  +-- rw mpls-tp-topology!

- TE Bandwidth Augmentations a described in section 3.1;
- TE Label Augmentations as described in section 3.2;

### 3.1. TE Bandwidth Augmentations

Following TE Bandwidth attributes are needed to be augmented to the module ietf-te-topology in [TE-Topology]:

- Augmentations for te-bandwidth in the max-link-bandwidth, max-resv-link-bandwidth and unreserved-bandwidth attributes of MPLS-TP TE Links are necessary for te-link-attributes and listed as follow. It is worth noting that for te-bandwidth in other places, this augment is not necessary.

```
  +--:(mpls-tp)
    +--rw mpls-tp-bandwidth?   uint64

  +--:(mpls-tp)
    +--rw mpls-tp-bandwidth?   uint64

  +--:(mpls-tp)
    +--rw mpls-tp-bandwidth?   uint64
```

- Augmentations for the max-lsp-bandwidth attribute are necessary for MPLS-TP TE Links and TTPs and listed as following. It is worth noting that for the other ‘max-lsp-bandwidth’, this augmentation is not necessary.

```
  +--:(mpls-tp)
    +--rw bandwidth-profile-name?   string
    +--rw bandwidth-profile-type?   identityref
    +--rw CIR?                      uint64
    +--rw EIR?                      uint64
    +--rw CBS?                      uint64
```
3.2. TE Label Augmentations

In MPLS-TP, the label allocation is done by NE, information about label values availability is not necessary to be provided to the controller. Moreover, MPLS-TP tunnels are currently established within a single domain.

Therefore this document does not define any MPLS-TP technology-specific augmentations, of the TE Topology model, for the TE label since no TE label related attributes should be instantiated for MPLS-TP Topologies.

4. Gap Analysis for MPLS-TP Tunnel Configuration

MPLS-TE technology-specific augmentations of the generic TE Tunnel model defined in [TE-MPLS].

This section analyses the minimum set of attributes that are required to be specified in an MPLS-TP technology-specific augmentation and not yet available in [TE-MPLS].

Additional attributes that may be required to support a broader set of MPLS-TP and/or MPLS-TE functions are for further study.

Although there are no guidance for augmentation in [TE-Tunnel], the following technology-specific augmentations need to be provided:

- TE Bandwidth Augmentations as described in section 4.1
- TE Label Augmentations as described in section 4.2
4.1. TE Bandwidth Augmentation

Following TE Bandwidth attributes are needed to be augmented for MPLS-TP to the to the module ietf-te in [TE-Tunnel], but are not yet defined in [TE-MPLS]:

- Augmentations for the te-bandwidth attribute of TE Tunnels under te/globals/tunnels are listed as follow. It is worth noting that for te-bandwidth in other places, this augmentation is not necessary.

```
augment /te:te/te:tunnels/te:tunnel/te:te-bandwidth/te:technology:
  +--:(mpls-tp)
    +--rw bandwidth-profile-name? string
    +--rw bandwidth-profile-type? identityref
    +--rw CIR? uint64
    +--rw EIR? uint64
    +--rw CBS? uint64
    +--rw EBS? uint64
```

4.2. TE Label Augmentation

Following TE Label attributes are needed to be augmented for MPLS-TP to the to the module ietf-te in [TE-Tunnel], but are not yet defined in [TE-MPLS]:

- Augmentations for the te-label attribute of MPLS-TP label hops are used to report the computed primary and secondary paths of MPLS-TP TE Tunnels as well as the route and the path of the MPLS-TP LSPs of the primary and secondary paths of MPLS-TP TE Tunnels. These augmentations are listed as follow, and it is worth noting for te-label in other places, there is no need to do the augmentation.

```
  +--:(mpls-tp)
    +--ro mpls-label? rt-types:mpls-label
```

```
  +--:(mpls-tp)
    +--ro mpls-label? rt-types:mpls-label
```
  +--:(mpls-tp)
    +--ro mpls-label?  rt-types:mpls-label

  +--:(mpls-tp)
    +--ro mpls-label?  rt-types:mpls-label

5. Related YANG Code

5.1. YANG Code for MPLS-TP Topology Augmentation

<CODE BEGINS>file "ietf-mpls-tp-topology@2019-03-11.yang"
module ietf-mpls-tp-topology {
  //yang-version 1.1;

  prefix "mpls-tp-topo";

  import ietf-network {
    prefix "nw";
  }

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

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

  import ietf-mpls-tp-types {
    prefix "mpls-tp-types";
  }

  organization
"Internet Engineering Task Force (IETF) TEAS WG";
contact

WG List: <mailto:teas@ietf.org>

ID-draft editor:
   Italo Busi (italo.busi@huawei.com);
   Haomian Zheng (zhenghaomian@huawei.com);
"

description
"This module defines technology-specific MPLS-TP topology
data model."

revision 2019-03-11 {
   description
   "version -00 as an I-D"
   reference
   "draft-busizheng-teas-mpls-tp-yang";
}

augment "/nw:networks/nw:network/nw:network-types/"
   + "tet:te-topology" {
      container mpls-tp-topology {
         presence "indicates a topology type of MPLS-TP layer.";
         description "mpls-tp te topology type";
      }
      description "augment network types to include mpls-tp
newtork";
   }

augment "/nw:networks/nw:network/nt:link/tet:te/"
   + "tet:te-link-attributes/"
   + "tet:interface-switching-capability/tet:max-lsp-bandwidth/"
   + "tet:te-bandwidth/tet:technology" {
      when "./././././././.nw:network-types/tet:te-topology/
         + "mpls-tp-topo:mpls-tp-topology" {
            description "MPLS-TP TE bandwidth.";
         }
      description "MPLS-TP bandwidth.";
      case mpls-tp {
         uses mpls-tp-types:mpls-tp-path-bandwidth;
      }
   }

augment "/nw:networks/nw:network/nt:termination-point/"
 + "tet:te/tet:interface-switching-capability/
 + "tet:max-lsp-bandwidth/tet:te-bandwidth/tet:technology"
{
 when "../../../../../../../nw:network-types/tet:te-topology/
 + "mpls-tp-topo:mpls-tp-topology" {
 description "Augment MPLS-TP TE bandwidth";
 }
 description "MPLS-TP bandwidth.";
 case mpls-tp {
   uses mpls-tp-types:mpls-tp-path-bandwidth;
 }
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
 + "tet:te-link-attributes/tet:max-link-bandwidth/
 + "tet:te-bandwidth/tet:technology" {
 when "../../../../../../nw:network-types/tet:te-topology/
 + "mpls-tp-topo:mpls-tp-topology" {
 description "MPLS-TP TE bandwidth.";
 }
 description "MPLS-TP bandwidth.";
 case mpls-tp {
   uses mpls-tp-types:mpls-tp-bandwidth;
 }
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
 + "tet:te-link-attributes/tet:max-resv-link-bandwidth/
 + "tet:te-bandwidth/tet:technology" {
 when "../../../../../../nw:network-types/tet:te-topology/
 + "mpls-tp-topo:mpls-tp-topology" {
 description "MPLS-TP TE bandwidth.";
 }
 description "MPLS-TP bandwidth.";
 case mpls-tp {
   uses mpls-tp-types:mpls-tp-bandwidth;
 }
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
 + "tet:te-link-attributes/tet:unreserved-bandwidth/
 + "tet:te-bandwidth/tet:technology" {
 when "../../../../../../nw:network-types/tet:te-topology/
 + "mpls-tp-topo:mpls-tp-topology" {
 description "MPLS-TP TE bandwidth.";
 }

5.2. YANG Code for MPLS-TP Tunnel Augmentation

```yml
augment "/te:te/te:tunnels/te:tunnel/" + "te:te-bandwidth/te:technology" {
  description "MPLS-TP bandwidth.";
  case mpls-tp {
    uses mpls-tp-types:mpls-tp-bandwidth;
  }
}
```

Busi & Zheng

Expires September 2019
description "MPLS-TP bandwidth."
  case mpls-tp {
    uses mpls-tp-types:mpls-tp-path-bandwidth;
  }
}

augment "/te:te/te:tunnels/te:tunnel/"
  + "te:p2p-primary-paths/te:p2p-primary-path/"
  + "te:computed-paths-properties/"
  + "te:computed-path-properties/"
  + "te:path-properties/te:path-route-objects/"
  + "te:path-computed-route-object/te:type/te:label/"
  + "te:label-hop/te:te-label/te:technology" {
  description "MPLS-TP label."
  case mpls-tp {
    uses mpls-tp-types:mpls-tp-path-label;
  }
  }

augment "/te:te/te:tunnels/te:tunnel/"
  + "te:p2p-primary-paths/te:p2p-primary-path/"
  + "te:lsp/"
  + "te:path-properties/te:path-route-objects/"
  + "te:path-computed-route-object/te:type/te:label/"
  + "te:label-hop/te:te-label/te:technology" {
  description "MPLS-TP label."
  case mpls-tp {
    uses mpls-tp-types:mpls-tp-path-label;
  }
  }

augment "/te:te/te:tunnels/te:tunnel/"
  + "te:p2p-secondary-paths/te:p2p-secondary-path/"
  + "te:computed-paths-properties/"
  + "te:computed-path-properties/"
  + "te:path-properties/te:path-route-objects/"
  + "te:path-computed-route-object/te:type/te:label/"
  + "te:label-hop/te:te-label/te:technology" {
  description "MPLS-TP label."
  case mpls-tp {
    uses mpls-tp-types:mpls-tp-path-label;
  }
  }

augment "/te:te/te:tunnels/te:tunnel/"
  + "te:p2p-secondary-paths/te:p2p-secondary-path/"
  + "te:lsp/"
5.3. MPLS-TP Specific YANG Types

<CODE BEGINS>file "ietf-mpls-tp-types@2019-03-11.yang"
module ietf-mpls-tp-types {
  prefix "mpls-tp-types";

  import ietf-routing-types {
    prefix "rt-types";
  }

  import ietf-eth-tran-types {
    prefix "etht-types";
  }

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

  ID-draft editor:
  Italo Busi (italo.busi@huawei.com);
  Haomian Zheng (zhenghaomian@huawei.com);

  description "This module defines technology-specific MPLS-TP types
  data model."
  revision 2019-03-11 {
    description "version -00 as an I-D";
    reference "draft-busizheng-teas-mpls-tp-yang";
  }

Busi & Zheng
Expires September 2019
grouping mpls-tp-path-bandwidth {
  description "Path bandwidth for MPLS-TP. ";
  leaf bandwidth-profile-name{
    type string;
    description "Name of Bandwidth Profile.";
  }
  leaf bandwidth-profile-type {
    type identityref {
      base etht-types:bandwidth-profile-type;
    }
    description "Type of Bandwidth Profile.";
  }
  leaf CIR {
    type uint64;
    description "Committed Information Rate in Kbps";
  }
  leaf EIR {
    type uint64;
    /* Need to indicate that EIR is not supported by RFC 2697
       must
       '../bw-profile-type = "etht-types:mef-10-bwp" or ' +
       '../bw-profile-type = "etht-types:rfc-2698-bwp" or ' +
       '../bw-profile-type = "etht-types:rfc-4115-bwp"/
       must
       '../bw-profile-type != "etht-types:rfc-2697-bwp"
     */
    description "Excess Information Rate in Kbps
     In case of RFC 2698, PIR = CIR + EIR";
  }
  leaf CBS {
    type uint64;
    description "Committed Burst Size in in KBytes";
  }
  leaf EBS {
    type uint64;
    description
"Excess Burst Size in KBytes.  
   In case of RFC 2698, PBS = CBS + EBS";

grouping mpls-tp-bandwidth {
   description "Bandwidth for MPLS-TP. ";
   leaf mpls-tp-bandwidth {
      type uint64{
         range "0..10000000000";
      }
      units "Kbps";
      description "Available bandwidth value expressed in kilobits per second";
   }
}

6. Open Issues

A few open issues are listed in this section for discussion with the WG experts:

- The value for ‘encoding’ in ietf-te-topology and ietf-te should be configured as ’lsp-encoding-packet’ for MPLS-TP;
- The value for ‘switching-type’ in ietf-te-topology and ietf-te should be configured as ’switching-pscl’ for MPLS-TP;
- There are still open issues for [TE-Tunnel], so the right directory may need to be confirmed up to the latest module ietf-te after maturity;
- Is it possible to integrate the proposal augmentation into [TE-MPLS]?
If the answer is ‘yes’, the following open issues need to address in the merged document.

- Some attributes will be needed to understand whether MPLS-TP specific features (such as no ECMP, no PHP, bidirectional LSP and GAL) are supported by the MPLS-TE topology and/or required to be supported by the MPLS-TE tunnel to be setup
- Per Tunnel-Termination-Point(TTP) modeling, one TTP per physical PE node should be sufficient for MPLS-TP;
- An empty container should be set for client-layer-adaption in the topology model for MPLS-TP;

Finally, it is not clear how to generate the inter-layer-lock-id for MPLS-TP and other layers, which may be considered in future.

7. Security

TBD.

8. Acknowledgements

We thank Loa Andersson and Igor Bryskin for providing useful suggestions for this draft.

9. References

9.1. Normative References


9.2. Informative References


Authors’ Addresses

Italo Busi
Huawei Technologies
Email: Italo.Busi@huawei.com

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com
A Yang Data Model for VN Operation

draft-ietf-teas-actn-vn-yang-05

Abstract

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

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on December 13, 2019.
Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with
respect to this document. Code Components extracted from this
document must include Simplified BSD License text as described in
Section 4.e of the Trust Legal Provisions and are provided without
warranty as described in the Simplified BSD License.

Table of Contents

Introduction.....................................................3
1........................................................................3
  1.1. Terminology...............................................3
  1.2. Tree diagram..............................................4
  1.3. Prefixes in Data Node Names...............................4
2. Use-case of VN Yang Model in the ACTN context..................5
  2.1. Type 1 VN....................................................5
  2.2. Type 2 VN....................................................6
3. High-Level Control Flows with Examples.........................7
  3.1. Type 1 VN Illustration....................................7
  3.2. Type 2 VN Illustration....................................9
4. VN Model Usage................................................12
  4.1. Customer view of VN......................................12
  4.2. Auto-creation of VN by MDSC............................12
  4.3. Innovative Services......................................12
    4.3.1. VN Compute..........................................12
    4.3.2. Multi-sources and Multi-destinations.................12
    4.3.3. Others..............................................13
    4.3.4. Summary.............................................14
5. VN YANG Model (Tree Structure)................................14
6. VN YANG Code..................................................16
7. JSON Example..................................................29
  7.1. VN JSON..................................................30
  7.2. TE-topology JSON.........................................35
8. Security Considerations.......................................51
9. IANA Considerations...........................................52
10. Acknowledgments..............................................53
11. References...................................................54
  11.1. Normative References....................................54
1. Introduction

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

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

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

- Characteristics of Access Points (APs) that describe customer’s end point characteristics;

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

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

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

Abstraction and Control of Traffic Engineered Networks (ACTN) describes a set of management and control functions used to operate one or more TE networks to construct virtual networks that can be represented to customers and that are built from abstractions of the underlying TE networks [RFC8453]. ACTN is the primary example of the usage of the VN Yang model.
Sections 2 and 3 provide the discussion of how the VN Yang model is applicable to the ACTN context where Virtual Network Service (VNS) operation is implemented for the Customer Network Controller (CNC)-Multi-Domain Service Coordinator (MSDC) interface (CMI).

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

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

1.1. Terminology

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

1.2. Tree diagram

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

1.3. Prefixes in Data Node Names

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

| Prefix | YANG module               | Reference     |
|--------+---------------------------+--------------|
| vn     | ietf-vn                   | [RFCXXXX]    |
| nw     | ietf-network              | [RFC8345]    |
| te-types | ietf-te-types          | [TE-Tunnel]  |
| te-topo | ietf-te-topology         | [TE-TOPO]    |

Table 1: Prefixes and corresponding YANG modules

Note: The RFC Editor will replace XXXX with the number assigned to the RFC once this draft becomes an RFC.
2. Use-case of VN Yang Model in the ACTN context

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

```
+-------+
|  CNC  |
+-------+
         |
|        |
|        |
|        |
| VN YANG + TE-topology YANG |
|        |
|        |
+---------------+
| MDSC          |
+---------------+

Figure 1. ACTN CMI
```

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

In the context of 5G transport application, 5G Traffic Provisioning Manager (TPM) that provides slicing requirements to the transport networks (i.e., MDSC) can be considered as a type of CNC. The ACTN CMI provides the necessary interface functions between 5G and transport networks in order to facilitate dynamic VN creation and its lifecycle management with proper feedback loop for monitoring.

2.1. Type 1 VN

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

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

If we were to create a VN where we have four VN-members as follows:
Where L1, L2, L3, L4, L7 and L8 correspond to a Customer End-Point, respectively.

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

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

Figure 2. Abstract Node (One node topology)

Modeling a VN as one abstract node is the easiest way for customers to express their end-to-end connectivity; however, customers are not limited to express their VN only with one abstract node. In some cases, more than one abstract nodes can be employed to express their VN.

2.2. Type 2 VN

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

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

```
+----------------------------------------------+
|             S1               S2              |
|              O---------------O               |
Lee & Dhody Expires December 2019 [Page 6]`
As you see from Figure 3, the Type 1 abstract node is depicted as a Type 1 abstract topology comprising of detailed virtual nodes and virtual links.

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

3. High-Level Control Flows with Examples

3.1. Type 1 VN Illustration

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

<table>
<thead>
<tr>
<th>VN-Member 1</th>
<th>L1-L4</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN-Member 2</td>
<td>L1-L7</td>
</tr>
<tr>
<td>VN-Member 3</td>
<td>L2-L4</td>
</tr>
<tr>
<td>VN-Member 4</td>
<td>L3-L8</td>
</tr>
</tbody>
</table>
Where L1, L2, L3, L4, L7 and L8 correspond to Customer End-Point, respectively.

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

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

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

```
+--------+       +--------+
  | CNC   |       | MDSC  |
+--------+       +--------+
     |                        |
     |                        |
CNC POST TE-topo model(with Conn. Matrix on one
|                      | Abstract node  |------------------------>
     POST /nw:networks/nw:network/
     nw:node/te-node-id/
     tet:connectivity-matrices/
     tet:connectivity-matrix
     HTTP 200

CNC POST the VN identifying AP, VNAP and VN-Members and maps to the TE-topo
| POST /VN |</--------
|          |           |
HTTP 200

CNC GET the VN YANG status
| GET /VN |
| HTTP 200 (VN with status: selected VN-members in case of multi s-d) |
|<------------------------|
```

If there is multi-dest’n module, then MDSC selects a src or dest’n and update VN YANG

Lee & Dhody Expires December 2019 [Page 8]
3.2. Type 2 VN Illustration

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

- VN-Member 1: L1-L4 (via S3, S4, and S5)
- VN-Member 2: L1-L7 (via S3, S4, S7, and S8)
- VN-Member 3: L2-L7 (via S9, S10, and S11)
- VN-Member 4: L3-L8 (via S9, S10, and S11)

Where the following topology is the underlay for Abstraction Node 1 (AN1).
There are two options depending on whether CNC or MDSC creates the single abstract node topology.

Case 1:

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

```
+--------+                                 +--------+
|  CNC   |                                |  MDSC  |
+--------+                                +--------+
|                                         |
|                                         |
CNC POST TE-topo model(with Conn. Abstract node and Explicit paths in The conn. Matrix) |
|<----------------------------------------| HTTP 200 |
| POST /nw:networks/nw:network/           |
|  nw:node/te-node-id/tet:connectivity-   |
|  matrices/tet:connectivity-matrix       |
|<----------------------------------------|

CNC POST the VN identifying AP, VNAP and VN-Members and maps to the TE-topo |
|<----------------------------------------| HTTP 200 |
| POST /VN                                |

CNC GET the VN YANG status |
|<----------------------------------------| HTTP 200 |
| GET /VN                                |

HTTP 200 (VN with status) |
|--------------------------|
```

Case 2:

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

```
+--------+                        +--------+
|  CNC   |                        |  MDSC  |
+--------+                        +--------+

CNC POST VN
Identifying AP, VNAP and VN-Members
POST /VN
--------------------------------> HTTP 200
<---------------------------------

CNC GET VN & POST TE-Topo Models (with Conn. Matrix on the Abstract Node and explicit paths in the Conn. Matrix)
GET /VN & POST /nw:networks/nw:network/
  nw:node/te-node-id/tet:
  connectivity-matrices/
  tet:connectivity-matrix
--------------------------------> HTTP 200
<---------------------------------

CNC GET the VN YANG status
GET /VN
----------------------------> HTTP 200 (VN with status)
<-------------------------------

MDSC populates a single Abst. node topology by itself

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

4.1. Customer view of VN

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

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

4.2. Auto-creation of VN by MDSC

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

4.3. Innovative Services

4.3.1. VN Compute

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

4.3.2. Multi-sources and Multi-destinations

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

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

4.3.3. Others

The VN Yang model can be easily augmented to support the mapping of VN to the Services such as L3SM and L2SM as described in [TE-MAP].

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

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

- Maintenance of AP and VNAP along with VN.
- VN construct to group of edge-to-edge links
- VN Compute (pre-instantiate)
- Multi-Source / Multi-Destination
- Ability to support various VN and VNS Types

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

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

5. VN YANG Model (Tree Structure)

module: ietf-vn
  +rw ap
    |   +rw access-point-list* [access-point-id]
    |     +rw access-point-id         uint32
    |     +rw access-point-name?     string
    |     +rw max-bandwidth?         te-types:te-bandwidth
    |     +rw avl-bandwidth?         te-types:te-bandwidth
    |     +rw vn-ap* [vn-ap-id]
    |       +rw vn-ap-id            uint32
    |     +rw vn?
    -> /vn/vn-list/vn-id
    |     +rw abstract-node?
    -> /nw:networks/network/node/tet:te-node-id
    |     +rw ltp?
    +rw vn
+-rw vn-list* [vn-id]
  +-rw vn-id              uint32
  +-rw vn-name?           string
  +-rw vn-topology-id?    te-types:te-topology-id
  +-rw abstract-node?
-\> /nw:networks/network/node/tet:te-node-id
  +-rw vn-member-list* [vn-member-id]
    |  +-rw vn-member-id            uint32
    |  +-rw src
    |     +-rw src?
-\> /ap/access-point-list/access-point-id
  |  +-rw src-vn-ap-id?
  |  |  +-rw multi-src? boolean {multi-src-dest}?
  |  +-rw dest
  |     +-rw dest?
-\> /ap/access-point-list/vn-ap/vn-ap-id
  |  +-rw multi-dest? boolean {multi-src-dest}?
  |  +-rw connectivity-matrix-id?
-\> /nw:networks/network/node/tet:te/node-attribute
/connectivity-matrices/connectivity-matrix/id
  |  +-ro oper-status? identityref
  |  +-ro if-selected? boolean {multi-src-dest}?
  +-rw admin-status? identityref
  +-ro oper-status? identityref
  +-rw vn-level-diversity? vn-disjointness

rpcs:
  +--x vn-compute
  +--w input
  |  +-w abstract-node?
-\> /nw:networks/network/node/tet:te-node-id
  |  +-w vn-member-list* [vn-member-id]
  |     |  +-w vn-member-id uint32
  |     |  +-w src
  |     |     +-w src?
-\> /ap/access-point-list/access-point-id
  |  +-w src-vn-ap-id?
  |  |  |  +-w multi-src? boolean {multi-src-dest}?
  |  |  +-w dest
  |  |     +-w dest?
-\> /ap/access-point-list/vn-ap/vn-ap-id
  |  |  |  +-w multi-dest? boolean {multi-src-dest}?
  |  |  +-w connectivity-matrix-id?
-\> /ap/access-point-list/access-point-id
  |  |  |  +-w dest-vn-ap-id?
6. VN YANG Code

The YANG code is as follows:

<CODE BEGINS> file "ietf-vn@2019-06-20.yang"

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

  /* Import network */
  import ietf-network {
    prefix "nw";
    reference
      "RFC 8345: A YANG Data Model for Network Topologies";
  }
}

<CODE ENDS>
/* Import network topology */
import ietf-network-topology {
    prefix "nt";
    reference
        "RFC 8345: A YANG Data Model for Network Topologies";
}

/* Import TE generic types */
import ietf-te-types {
    prefix "te-types";
    reference
        "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
}

/* Import Abstract TE Topology */
import ietf-te-topology {
    prefix "tet";
    reference
        "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic Engineering (TE) Topologies";
}

organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS) Working Group";
contact
    "Editor: Young Lee <youngleex.tx@gmail.com>
      : Dhruv Dhody <dhruv.ietf@gmail.com>";
description
    "This module contains a YANG module for the VN. It describes a VN operation module that takes place in the context of the CNC-MDSC Interface (CMI) of the ACTN architecture where the CNC is the actor of a VN Instantiation/modification/deletion.";
revision 2019-06-10 {
    description
        "initial version.";
    reference
        "TBD";
}
/*
 * Features
 */
feature multi-src-dest {
description
   "Support for selection of one src or destination
   among multiple.";
}

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

identity vn-state-type {
   description
      "Base identity for VN state";
}
identity vn-state-up {
   base vn-state-type;
   description "VN state up";
}
identity vn-state-down {
   base vn-state-type;
   description "VN state down";
}
identity vn-admin-state-type {
   description
      "Base identity for VN admin states";
}
identity vn-admin-state-up {
   base vn-admin-state-type;
   description "VN administratively state up";
}
identity vn-admin-state-down {
   base vn-admin-state-type;
   description "VN administratively state down";
}
identity vn-compute-state-type {
    description
        "Base identity for compute states";
}

identity vn-compute-state-computing {
    base vn-compute-state-type;
    description
        "State path compute in progress";
}

identity vn-compute-state-computation-ok {
    base vn-compute-state-type;
    description
        "State path compute successful";
}

identity vn-compute-state-computation-failed {
    base vn-compute-state-type;
    description
        "State path compute failed";
}

/*
 * Groupings
 */

typedef vn-disjointness {
    type bits {
        bit node {
            position 0;
            description "node disjoint";
        }
        bit link {
            position 1;
            description "link disjoint";
        }
        bit srlg {
            position 2;
            description "srlg disjoint";
        }
    }
    description
        "type of the resource disjointness for
        VN level applied across all VN members
        in a VN";
}

grouping vn-ap {
description
  "VNAP related information";
leaf vn-ap-id {
  type uint32;
  description
    "unique identifier for the referred VNAP";
}
leaf vn {
  type leafref {
    path "/vn/vn-list/vn-id";
  }
  description
    "reference to the VN";
}
leaf abstract-node {
  type leafref {
    path "/nw:networks/nw:network/nw:node/
      +tet:te-node-id";
  }
  description
    "a reference to the abstract node in TE Topology";
}
leaf ltp {
  type leafref {
    path "/nw:networks/nw:network/nw:node/
      +nt:termination-point/tet:te-tp-id";
  }
  description
    "Reference LTP in the TE-topology";
}
}
grouping access-point{
  description
    "AP related information";
  leaf access-point-id {
    type uint32;
    description
      "unique identifier for the referred access point";
  }
  leaf access-point-name {
    type string;
    description

leaf max-bandwidth {
    type te-types:te-bandwidth;
    description "max bandwidth of the AP";
}
leaf avl-bandwidth {
    type te-types:te-bandwidth;
    description "available bandwidth of the AP";
}
/*add details and any other properties of AP, not associated by a VN
CE port, PE port etc.*/
list vn-ap {
    key vn-ap-id;
    uses vn-ap;
    description "list of VNAP in this AP";
}
} //access-point

grouping vn-member {
    description "vn-member is described by this container";
    leaf vn-member-id {
        type uint32;
        description "vn-member identifier";
    }
    container src {
        description "the source of VN Member";
        leaf src {
            type leafref {
                path "/ap/access-point-list/access-point-id";
            }
            description "reference to source AP";
        }
        leaf src-vn-ap-id {
            type leafref {
                path "/ap/access-point-list/access-point-id";
            }
            description "reference to VNAP";
        }
    }
}
leaf multi-src {
    if-feature multi-src-dest;
    type boolean;
    description
        "Is source part of multi-source, where only one of the source is enabled";
}

container dest {
    description
        "the destination of VN Member";
    leaf dest {
        type leafref {
            path "/ap/access-point-list/access-point-id";
        }
        description
            "reference to destination AP";
    }
    leaf dest-vn-ap-id{
        type leafref {
            path "/ap/access-point-list/vn-ap/vn-ap-id";
        }
        description
            "reference to dest VNA";
    }
    leaf multi-dest {
        if-feature multi-src-dest;
        type boolean;
        description
            "Is destination part of multi-destination, where only one of the destination is enabled";
    }
}

leaf connectivity-matrix-id{
    type leafref {
    }
}
grouping policy {
    description
    "policy related to vn-member-id";
    leaf local-reroute {
        type boolean;
        description
        "Policy to state if reroute can be done locally";
    }
    leaf push-allowed {
        type boolean;
        description
        "Policy to state if changes can be pushed to the customer";
    }
    leaf incremental-update {
        type boolean;
        description
        "Policy to allow only the changes to be reported";
    }
}

/*
   grouping vn-policy {
       description
       "policy for VN-level diversity";
       leaf vn-level-diversity {
           type vn-disjointness;
           description
           "the type of disjointness on the VN level (i.e., across all VN members)";
       }
   }
   */

/*
   grouping metrics-op {
       description
       "metric related information";
       list metric{
           key "metric-type";
           description
           "metric related information";
           key "metric-value";
           type string;
           description
           "The value of the metric";
       }
   }
*/
config false;
description
   "The list of metrics for VN";
leaf metric-type {
    type identityref {
        base te-types:path-metric-type;
    }
    description
        "The VN metric type.";
}
leaf value{
    type uint32;
    description
        "The limit value";
}
}
*/
/*
grouping metrics {
    description
        "metric related information";
    list metric{
        key "metric-type";
        description
            "The list of metrics for VN";
        uses te:path-metrics-bounds_config;
        container optimize{
            description
                "optimizing constraints";
            leaf enabled{
                type boolean;
                description
                    "Metric to optimize";
            }
            leaf value{
                type uint32;
                description
                    "The computed value";
            }
        }
    }
}
*/
/*
grouping service-metric {
  description "service-metric";
  uses te:path-objective-function_config;
  uses metrics;
  uses te-types:common-constraints_config;
  uses te:protection-restoration-params_config;
  uses policy;
}//service-metric
*/
/*
* Configuration data nodes
*/

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

container vn {
  description "VN configurations";
  list vn-list {
    key "vn-id";
    description "a virtual network is identified by a vn-id";
    leaf vn-id {
      type uint32;
      description "a unique vn identifier";
    }
    leaf vn-name {
      type string;
      description "vn name";
    }
    leaf vn-topology-id{
type te-types:te-topology-id;

description
"An optional identifier to the TE Topology Model where the abstract nodes and links of the Topology can be found for Type 2 VNS";
}

leaf abstract-node {

type leafref {
    path "/nw:networks/nw:network/nw:node/
    + "tet:te-node-id";
}

description
"a reference to the abstract node in TE Topology";
}

list vn-member-list{
    key "vn-member-id";
    description
"List of VN-members in a VN";
    uses vn-member;
    /*uses metrics-op;*/
    leaf oper-status {
        type identityref {
            base vn-state-type;
        }
        config false;
        description
"VN-member operational state.";
    }
}

leaf if-selected{
    if-feature multi-src-dest;
    type boolean;
    default false;
    config false;
    description
"Is the vn-member is selected among the multi-src/dest options";
}

/*
container multi-src-dest{
    if-feature multi-src-dest;
    config false;
}*/
description "The selected VN Member when multi-src and/or multi-destination is enabled.";
leaf selected-vn-member{
  type leafref {
    path "+//vn/vn-list/vn-member-list" + "+//vn-member-id";
  }
  description "The selected VN Member along the set of source and destination configured with multi-source and/or multi-destination";
}
/*
/*uses service-metric;*/
leaf admin-status {
  type identityref {
    base vn-admin-state-type;
  }
  default vn-admin-state-up;
  description "VN administrative state.";
}
leaf oper-status {
  type identityref {
    base vn-state-type;
  }
  config false;
  description "VN operational state.";
}
uses vn-policy;
}//vn

/*
* Notifications - TBD
*/
/*
* RPC
*/
rpc vn-compute{
  description "The VN computation without actual instantiation";
  input {

leaf abstract-node {
  type leafref {
    path "/nw:networks/nw:network/nw:node/" + "tet:te-node-id";
  }
  description "a reference to the abstract node in TE Topology";
}
list vn-member-list{
  key "vn-member-id";
  description "List of VN-members in a VN";
  uses vn-member;
}
uses vn-policy;
/*uses service-metric;*/
}
output {
  list vn-member-list{
    key "vn-member-id";
    description "List of VN-members in a VN";
    uses vn-member;
    leaf if-selected{
      if-feature multi-src-dest;
      type boolean;
      default false;
      description "Is the vn-member is selected among the multi-src/dest options";
    }
    /*uses metrics-op;*/
    leaf compute-status {
      type identityref {
        base vn-compute-state-type;
      }
      description "VN-member compute state.";
    }
  }
  /*uses metrics-op;*/
  leaf compute-status {
    type identityref {
      base vn-compute-state-type;
    }
    description "VN-member compute state.";
  }
  /*
  container multi-src-dest{
    if-feature multi-src-dest;
    description
  }
  */
"The selected VN Member when multi-src
and/or multi-destination is enabled.");
leaf selected-vn-member-id{
    type uint32;
description
    "The selected VN Member-id from the
    input";
}
*/

7. JSON Example

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

The example in this section includes following VN

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

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

- **VN3 (Type 1):** This VN has a multi-source, multi-destination
  feature enable for VN Member 104 (L1 to L4)/107 (L1 to L7)
  [multi-sr] and VN Member 204 (L2 to L4)/304 (L3 to L4) [multi-
  dest] usecase. The selected VN-member is known via the field "if-
  selected" and the corresponding connectivity-matrix-id.
Note that the VN YANG model also include the AP and VNAP which shows various VN using the same AP.

7.1. VN JSON

```json
{
    "ap": {
        "access-point-list": [
            {
                "access-point-id": 101,
                "access-point-name": "101",
                "vn-ap": [
                    {
                        "vn-ap-id": 10101,
                        "vn": 1,
                        "abstract-node": "D1",
                        "ltp": "1-0-1"
                    },
                    {
                        "vn-ap-id": 10102,
                        "vn": 2,
                        "abstract-node": "D2",
                        "ltp": "1-0-1"
                    },
                    {
                        "vn-ap-id": 10103,
                        "vn": 3,
                        "abstract-node": "D3",
                        "ltp": "1-0-1"
                    }
                ]
            },
            {
                "access-point-id": 202,
                "access-point-name": "202",
                "vn-ap": [
                    {
                        "vn-ap-id": 20201,
                        "vn": 1,
                        "abstract-node": "D1",
                        "ltp": "2-0-2"
                    }
                ]
            },
            {
                "access-point-id": 303,
                "access-point-name": "303",
                "vn-ap": [
                    {
                        "vn-ap-id": 30301,
```
"vn": 1,
"abstract-node": "D1",
"ltp": "3-0-3"
},
{
"vn-ap-id": 30303,
"vn": 3,
"abstract-node": "D3",
"ltp": "3-0-3"
}
},
{
"access-point-id": 440,
"access-point-name": "440",
"vn-ap": [
{
"vn-ap-id": 44001,
"vn": 1,
"abstract-node": "D1",
"ltp": "4-4-0"
}
]
},
{
"access-point-id": 550,
"access-point-name": "550",
"vn-ap": [
{
"vn-ap-id": 55002,
"vn": 2,
"abstract-node": "D2",
"ltp": "5-5-0"
}
]
},
{
"access-point-id": 770,
"access-point-name": "770",
"vn-ap": [
{
"vn-ap-id": 77001,
"vn": 1,
"abstract-node": "D1",
"ltp": "7-7-0"
}
],
{
"vn-ap-id": 77003,
"vn": 3,
"abstract-node": "D3",
"ltp": "7-7-0"
}
),
{
"access-point-id": 880,
"access-point-name": "880",
"vn-ap": [
{
"vn-ap-id": 88001,
"vn": 1,
"abstract-node": "D1",
"ltp": "8-8-0"
},
{
"vn-ap-id": 88003,
"vn": 3,
"abstract-node": "D3",
"ltp": "8-8-0"
}
]
}
},
"vn":{
"vn-list": [
{
"vn-id": 1,
"vn-name": "vn1",
"vn-topology-id": "te-topology:abstract1",
"abstract-node": "D1",
"vn-member-list": [
{
"vn-member-id": 104,
"src": {
"src": 101,
"src-vn-ap-id": 10101,
},
"dest": {
"dest": 440,
"dest-vn-ap-id": 44001,
},
"connectivity-matrix-id": 104
},
{
"vn-member-id": 107,
"src": {
"src": 101,
"src-vn-ap-id": 10101,
},
"dest": {
    "dest": 770,
    "dest-vn-ap-id": 77001,
},
"connectivity-matrix-id": 107
},
{ "vn-member-id": 204,
"src": {
    "src": 202,
    "dest-vn-ap-id": 20401,
},
"dest": {
    "dest": 440,
    "dest-vn-ap-id": 44001,
},
"connectivity-matrix-id": 204
},
{ "vn-member-id": 308,
"src": {
    "src": 303,
    "src-vn-ap-id": 30301,
},
"dest": {
    "dest": 880,
    "src-vn-ap-id": 88001,
},
"connectivity-matrix-id": 308
},
{ "vn-member-id": 108,
"src": {
    "src": 101,
    "src-vn-ap-id": 10101,
},
"dest": {
    "dest": 880,
    "dest-vn-ap-id": 88001,
},
"connectivity-matrix-id": 108
}
}
"vn-id": 2,
"vn-name": "vn2"
"vn-topology-id": "te-topology:abstract2",
"abstract-node": "D2",
"vn-member-list": [
  {
    "vn-member-id": 105,
    "src": {
      "src": 101,
      "src-vn-ap-id": 10102,
    },
    "dest": {
      "dest": 550,
      "dest-vn-ap-id": 55002,
    },
    "connectivity-matrix-id": 105
  }
],
{
  "vn-id": 3,
  "vn-name": "vn3",
  "vn-topology-id": "te-topology:abstract3",
  "abstract-node": "D3",
  "vn-member-list": [
    {
      "vn-member-id": 104,
      "src": {
        "src": 101,
      },
      "dest": {
        "dest": 440,
        "multi-dest": true
      }
    },
    {
      "vn-member-id": 107,
      "src": {
        "src": 101,
        "src-vn-ap-id": 10103,
      },
      "dest": {
        "dest": 770,
        "dest-vn-ap-id": 77003,
        "multi-dest": true
      },
      "connectivity-matrix-id": 107,
      "if-selected": true,
    },
    {
      "vn-member-id": 204,
7.2. TE-topology JSON

```json
{
    "networks": {
        "network": {
            "network-types": {
                "te-topology": {}
            },
            "network-id": "abstract1",
            "provider-id": 201,
            "client-id": 600,
            "te-topology-id": "te-topology:abstract1",
            "node": {
                "node-id": "D1",
                "te-node-id": "2.0.1.1",
                "te": {
```
"te-node-attributes": {
    "domain-id": 1,
    "is-abstract": [null],
    "connectivity-matrices": {
        "is-allowed": true,
        "path-constraints": {
            "bandwidth-generic": {
                "te-bandwidth": {
                    "generic": [
                        {
                            "generic": "0x1p10",
                        }
                    ]
                }
            }
        }
    }
},
"disjointness": "node link srlg",
"connectivity-matrix": [
    {
        "id": 104,
        "from": "1-0-1",
        "to": "4-4-0"
    },
    {
        "id": 107,
        "from": "1-0-1",
        "to": "7-7-0"
    },
    {
        "id": 204,
        "from": "2-0-2",
        "to": "4-4-0"
    },
    {
        "id": 308,
        "from": "3-0-3",
        "to": "8-8-0"
    },
    {
        "id": 108,
        "from": "1-0-1",
        "to": "8-8-0"
    }
],
"termination-point": [


"tp-id": "1-0-1",
"te-tp-id": 10001,
"te": {
   "interface-switching-capability": [ 
   {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
   }
   ]
},
},

"tp-id": "1-1-0",
"te-tp-id": 10100,
"te": {
   "interface-switching-capability": [ 
   {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
   }
   ]
},
},

"tp-id": "2-0-2",
"te-tp-id": 20002,
"te": {
   "interface-switching-capability": [ 
   {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
   }
   ]
},
},

"tp-id": "2-2-0",
"te-tp-id": 20200,
"te": {
   "interface-switching-capability": [ 
   {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
   }
   ]
},
}
"tp-id": "3-0-3",
"te-tp-id": 30003,
"te": {
   "interface-switching-capability": [
      {
         "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
},

"tp-id": "3-3-0",
"te-tp-id": 30300,
"te": {
   "interface-switching-capability": [
      {
         "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
},

"tp-id": "4-0-4",
"te-tp-id": 40004,
"te": {
   "interface-switching-capability": [
      {
         "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
},

"tp-id": "4-4-0",
"te-tp-id": 40400,
"te": {
   "interface-switching-capability": [
      {
         "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
},

"tp-id": "5-0-5",
"te-tp-id": 50005,
"te-tp-id": 50005,
"te": {
    "interface-switching-capability": [{
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
    ]
},
} {
"tp-id": "5-5-0",
"te-tp-id": 50500,
"te": {
    "interface-switching-capability": [{
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
    ]
},
} {
"tp-id": "6-0-6",
"te-tp-id": 60006,
"te": {
    "interface-switching-capability": [{
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
    ]
},
} {
"tp-id": "6-6-0",
"te-tp-id": 60600,
"te": {
    "interface-switching-capability": [{
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
    ]
},
} {
"tp-id": "7-0-7",
"te-tp-id": 70007,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

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

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

{ "tp-id": "8-8-0",
  "te-tp-id": 80800,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
{
    "network-types": {
        "te-topology": {}
    },
    "network-id": "abstract2",
    "provider-id": 201,
    "client-id": 600,
    "te-topology-id": "te-topology:abstract2",
    "node": [
        {
            "node-id": "D2",
            "te-node-id": "2.0.1.2",
            "te": {
                "te-node-attributes": {
                    "domain-id": 1,
                    "is-abstract": [null],
                    "connectivity-matrices": {
                        "is-allowed": true,
                        "underlay": {
                            "enabled": true
                        },
                        "path-constraints": {
                            "bandwidth-generic": {
                                "te-bandwidth": {
                                    "generic": [
                                        {
                                            "generic": "0x1p10"
                                        }
                                    ]
                                }
                            }
                        },
                        "optimizations": {
                            "objective-function": {
                                "objective-function-type": "of-maximize-residual-bandwidth"
                            }
                        }
                    }
                }
            },
            "connectivity-matrix": [
                {
                    "id": 105,
                    "from": "1-0-1",
                    "to": "5-5-0",
                    "underlay": {
                        "enabled": true,
                        "primary-path": {
                            "network-ref": "absolute",
                            "path-element": [
                                {
                                    "generic": "0x1p10"
                                }
                            ]
                        }
                    }
                }
            ]
        }
    ]
}
"path-element-id": 1,
"index": 1,
"numbered-hop": {
"address": "4.4.4.4",
"hop-type": "STRICT"
}
},
{"path-element-id": 2,
"index": 2,
"numbered-hop": {
"address": "7.7.7.7",
"hop-type": "STRICT"
}
}]
"termination-point": [
{"tp-id": "1-0-1",
"te-tp-id": 10001,
"te": {
"interface-switching-capability": [
{"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}],
{"tp-id": "1-1-0",
"te-tp-id": 10100,
"te": {
"interface-switching-capability": [
{"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]
}],
{"tp-id": "1-0-0-1",
"te-tp-id": 10001,
"te": {
"interface-switching-capability": [
{"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
]}
}
"tp-id": "2-0-2",
"te-tp-id": 20002,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "2-2-0",
"te-tp-id": 20200,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "3-0-3",
"te-tp-id": 30003,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "3-3-0",
"te-tp-id": 30300,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "4-0-4",
"te-tp-id": 20002,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "4-2-0",
"te-tp-id": 20200,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "4-3-0",
"te-tp-id": 30003,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{"tp-id": "4-3-0",
"te-tp-id": 30300,
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
}
"te-tp-id": 40004,
"te": {  
"interface-switching-capability": [  
{  
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}  
]  
},
"tp-id": "4-4-0",
"te-tp-id": 40400,
"te": {  
"interface-switching-capability": [  
{  
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}  
]  
},
"tp-id": "5-0-5",
"te-tp-id": 50005,
"te": {  
"interface-switching-capability": [  
{  
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}  
]  
},
"tp-id": "5-5-0",
"te-tp-id": 50500,
"te": {  
"interface-switching-capability": [  
{  
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}  
]  
},
"tp-id": "6-0-6",
"te-tp-id": 60006,
"te": {
   "interface-switching-capability": [
      {
         "switching-capability": "switching-otn",
         "encoding": "lsp-encoding-oduk"
      }
   ]
},

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

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

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

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

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

"network-types": { 
    "te-topology": { } 
},
"network-id": "abstract3",
"provider-id": 201,
"client-id": 600,
"te-topology-id": "te-topology:abstract3",
"node": [ 
    { 
        "node-id": "D3",
        "te-node-id": "3.0.1.1",
        "te": { 
            "te-node-attributes": { 
                "domain-id": 3,
                "is-abstract": [null],
                "connectivity-matrices": { 
                    "is-allowed": true,
                    "path-constraints": { 
                        "bandwidth-generic": { 
                            "te-bandwidth": { 
                                "generic": [ 
                                    "generic": "0x1p10",
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
]
"connectivity-matrix": [
  {
    "id": 107,
    "from": "1-0-1",
    "to": "7-7-0"
  },
  {
    "id": 308,
    "from": "3-0-3",
    "to": "8-8-0"
  }
],
"termination-point": [
  {
    "tp-id": "1-0-1",
    "te-tp-id": 10001,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-1-0",
    "te-tp-id": 10100,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "2-0-2",
    "te-tp-id": 20002,
    "te": {
      "interface-switching-capability": [}
{ "switching-capability": "switching-otn", "encoding": "lsp-encoding-oduk" }
}
}

{ "tp-id": "2-2-0", "te-tp-id": 20200, "te": { "interface-switching-capability": [ { "switching-capability": "switching-otn", "encoding": "lsp-encoding-oduk" } ] }
}

{ "tp-id": "3-0-3", "te-tp-id": 30003, "te": { "interface-switching-capability": [ { "switching-capability": "switching-otn", "encoding": "lsp-encoding-oduk" } ] }
}

{ "tp-id": "3-3-0", "te-tp-id": 30300, "te": { "interface-switching-capability": [ { "switching-capability": "switching-otn", "encoding": "lsp-encoding-oduk" } ] }
}

{ "tp-id": "4-0-4", "te-tp-id": 40004, "te": { "interface-switching-capability": [ { "switching-capability": "switching-otn", "encoding": "lsp-encoding-oduk" } ] }
}
"switching-capability": "switching-otn",
"encoding": "lsp-encoding-oduk"
}
}
{
  "tp-id": "5-0-5",
  "te-tp-id": 50005,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
{
  "tp-id": "5-5-0",
  "te-tp-id": 50500,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
{
  "tp-id": "6-0-6",
  "te-tp-id": 60006,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
"encoding": "lsp-encoding-oduk"
}
]
},
{
  "tp-id": "6-6-0",
  "te-tp-id": 60600,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "7-0-7",
  "te-tp-id": 70007,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "7-7-0",
  "te-tp-id": 70700,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "8-0-8",
  "te-tp-id": 80008,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
8. Security Considerations

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

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

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

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

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

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

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

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

- vn-member-id
  - src
  - src-vn-ap-id
  - dest
  - dest-vn-ap-id
  - connectivity-matrix-id

9. IANA Considerations

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

Lee & Dhody Expires December 2019
This document registers the following YANG modules in the YANG Module

Names registry [RFC6020]:

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

10. Acknowledgments

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

11.1. Normative References


11.2. Informative References


12. Contributors

Contributor’s Addresses

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

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

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

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

Authors’ Addresses

Young Lee (ed.)
Futurewei Technologies
Email: younglee.tx@gmail.com

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

Igor Bryskin  
Huawei  
Email: ibryskin@futurewei.com

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

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

Peter Park  
KT  
Email: peter.park@kt.com
Applicability of YANG models for Abstraction and Control of Traffic Engineered Networks
draft-ietf-teas-actn-yang-03

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt
Internet-Draft                ACTN YANG                  February 2019

The list of Internet-Draft Shadow Directories can be accessed at

This Internet-Draft will expire on August 23, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document.  Please review these documents
carefully, as they describe your rights and restrictions with
respect to this document.  Code Components extracted from this
document must include Simplified BSD License text as described in
Section 4.e of the Trust Legal Provisions and are provided without
warranty as described in the Simplified BSD License.

Abstract

Abstraction and Control of TE Networks (ACTN) refers to the set of
virtual network operations needed to orchestrate, control and manage
large-scale multi-domain TE networks, so as to facilitate network
programmability, automation, efficient resource sharing, and end-to-
end virtual service aware connectivity and network function
virtualization services.

This document explains how the different types of YANG models
defined in the Operations and Management Area and in the Routing
Area are applicable to the ACTN framework. This document also shows
how the ACTN architecture can be satisfied using classes of data
model that have already been defined, and discusses the
applicability of specific data models that are under development. It
also highlights where new data models may need to be developed.

Table of Contents

1. Introduction ................................................ 3
2. Abstraction and Control of TE Networks (ACTN) Architecture .. 3
3. Service Models .............................................. 5
4. Service Model Mapping to ACTN ............................... 7
1. Introduction

Abstraction and Control of TE Networks (ACTN) describes a method for operating a Traffic Engineered (TE) network (such as an MPLS-TE network or a layer 1 transport network) to provide connectivity and virtual network services for customers of the TE network. The services provided can be tuned to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers. More details about ACTN can be found in Section 2.

Data models are a representation of objects that can be configured or monitored within a system. Within the IETF, YANG [RFC6241] is the language of choice for documenting data models, and YANG models have been produced to allow configuration or modelling of a variety of network devices, protocol instances, and network services. YANG data models have been classified in [RFC8199] and [RFC8309].

This document shows how the ACTN architecture can be satisfied using various classes of data model that have already been defined, and discusses the applicability of specific data models that are under development. It also highlights where new data models may need to be developed.

2. Abstraction and Control of TE Networks (ACTN) Architecture

[RFC8453] describes the architecture model for ACTN including the entities (Customer Network Controller (CNC), Multi-domain Service...
Coordinator (MDSC), and Provisioning Network Controller (PNC)) and their interfaces.

Figure 1 depicts a high-level control and interface architecture for ACTN and is a reproduction of Figure 3 from [RFC8453]. A number of key ACTN interfaces exist for deployment and operation of ACTN-based networks. These are highlighted in Figure 1 (ACTN Interfaces) below:

---

The interfaces and functions are described below (without modifying the definitions) in [RFC8453]:

---
The CNC-MDSC Interface (CMI) is an interface between a CNC and an MDSC. This interface is used to communicate the service request or application demand. A request will include specific service properties, for example, services type, bandwidth and constraint information. These constraints SHOULD be measurable by MDSC and therefore visible to CNC via CMI. The CNC can also request the creation of the virtual network service based on underlying physical resources to provide network services for the applications. The CNC can provide the end-point information/characteristics together with traffic matrix specifying specific customer constraints. The MDSC may also report potential network topology availability if queried for current capability from the Customer Network Controller. Performance monitoring is also applicable in CMI, which enables the MDSC to report network parameters/telemetries that may guide the CNC to create/change their services.

The MDSC-PNC Interface (MPI) is an interface between a MDSC and a PNC. It allows the MDSC to communicate requests to create/delete connectivity or to modify bandwidth reservations in the physical network. In multi-domain environments, each PNC is responsible for a separate domain. The MDSC needs to establish multiple MPIS, one for each PNC and perform coordination between them to provide cross-domain connectivity. MPI plays an important role for multi-vendor mechanism, interoperability can be achieved by standardized interface modules.

The South-Bound Interface (SBI) is the provisioning interface for creating forwarding state in the physical network, requested via the PNC. The SBI is not in the scope of ACTN, however, it is included in this document so that it can be compared to models in [RFC8309].

3. Service Models

[RFC8309] introduces a reference architecture to explain the nature and usage of service YANG models in the context of service orchestration. Figure 2 below depicts this relationship and is a reproduction of Figure 2 from [RFC8309]. Four models depicted in Figure 2 are defined as follows:

Customer Service Model: A customer service model is used to describe a service as offer or delivered to a customer by a network operator.
Service Delivery Model: A service delivery model is used by a network operator to define and configure how a service is provided by the network.
Network Configuration Model: A network configuration model is used by a network orchestrator to provide network-level configuration model to a controller.

Device Configuration Model: A device configuration model is used by a controller to configure physical network elements.

---

Figure 2: An SDN Architecture with a Service Orchestrator
4. Service Model Mapping to ACTN

YANG models coupled with the RESTCONF/NETCONF protocol [RFC6241][RFC8040] provides solutions for the ACTN framework. This section explains which types of YANG models apply to each of the ACTN interfaces.

Refer to Figure 5 of [RFC8453] for details of the mapping between ACTN functions and service models. In summary, the following mappings are held between Service Yang Models in [RFC8309] and the ACTN interfaces in [RFC8453].

- Customer Service Model <-> CMI
- Network Configuration Model <-> MPI
- Device Configuration Model <-> SBI

4.1. Customer Service Models in the ACTN Architecture (CMI)

Customer Service Models, which are used between a customer and a service orchestrator as in [RFC8309], should be used between the CNC and MDSC (e.g., CMI) serving as providing a simple intent-like model/interface.

Among the key functions of Customer Service Models on the CMI is the service request. A request will include specific service properties, including: service type and its characteristics, bandwidth, constraint information, and end-point characteristics.

The following table provides a list of functions needed to build the CMI. They are mapped with Customer Service Models.

<table>
<thead>
<tr>
<th>Function</th>
<th>Yang Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN Service Request</td>
<td>[ACTN-VN-YANG]</td>
</tr>
<tr>
<td>VN Computation Request</td>
<td>[ACTN-VN-YANG]*</td>
</tr>
<tr>
<td>TE &amp; Service Mapping</td>
<td>[TE-Service-Mapping]**</td>
</tr>
<tr>
<td>VN Performance Monitoring Telemetry</td>
<td>[ACTN-PM-Telemetry]***</td>
</tr>
<tr>
<td>Topology Abstraction</td>
<td>[TE-topology]****</td>
</tr>
<tr>
<td>Layer 1 Connectivity Service Model</td>
<td>[L1CSM]</td>
</tr>
<tr>
<td>Layer 2 VPN Service Model</td>
<td>[RFC8466]</td>
</tr>
<tr>
<td>Layer 3 VPN Service Model</td>
<td>[RFC8299]</td>
</tr>
</tbody>
</table>
VN computation request in the CMI context means network path computation request based on customer service connectivity request constraints prior to the instantiation of a VN creation.

[TE-Service-Mapping] provides a mapping and cross-references between service models (e.g., L3SM, L2SM, L1CSM) and TE model via [ACTN-VN-YANG] and [TE-topology]. This model can be used as either Customer Service Models, or Service Delivery model described in Section 4.2.

[ietf-actn-te-kpi-telemetry model in [ACTN-PM-Telemetry] describes performance telemetry for ACTN VN model. This module also allows autonomic traffic engineering scaling intent configuration mechanism on the VN level. Scale in/out criteria might be used for network autonomies in order the controller to react to a certain set of variations in monitored parameters. Moreover, this module also provides mechanism to define aggregated telemetry parameters as a grouping of underlying VN level telemetry parameters.

[TE-Topology’s Connectivity Matrices/Matrix construct can be used to instantiate VN Service via a suitable referencing and mapping with [ACTN-VN-YANG]].

4.2. Service Delivery Models in ACTN Architecture

The Service Delivery Models where the service orchestration and the network orchestration could be implemented as separate components as seen in [RFC8309]. On the other hand, from an ACTN architecture point of view, the service delivery model between the service orchestrator and the network orchestrator is an internal interface between sub-components of the MDSC in a single MDSC model.

In the MDSC hierarchical model where there are multiple MDSCs, the interface between the top MDSC and the bottom MDSC can be mapped to service delivery models.

4.3. Network Configuration Models in ACTN Architecture (MPI)

The Network Configuration Models is used between the network orchestrator and the controller in [RFC8309]. In ACTN, this model is used primarily between a MDSC and a PNC. The Network Configuration Model can be also used for the foundation of more advanced models, like hierarchical MDSCs (see Section 4.5)

The Network Configuration Model captures the parameters which are network wide information.
The following table provides a list of functions needed to build the MPI. They are mapped with Network Configuration Yang Models. Note that various Yang models are work in progress.

<table>
<thead>
<tr>
<th>Function</th>
<th>Yang Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Scheduling</td>
<td>[Schedule]</td>
</tr>
<tr>
<td>Path computation</td>
<td>[PATH_COMPUTATION-API]</td>
</tr>
<tr>
<td>Tunnel/LSP Provisioning</td>
<td>[TE-tunnel]</td>
</tr>
<tr>
<td>Topology Abstraction</td>
<td>[TE-topology]</td>
</tr>
<tr>
<td>Client Signal Description</td>
<td>[Client-signal]</td>
</tr>
<tr>
<td>Service Provisioning</td>
<td>[Client-signal]&amp;[TE-tunnel]*</td>
</tr>
<tr>
<td>OTN Topology Abstraction</td>
<td>[OTN-topo]</td>
</tr>
<tr>
<td>WSON Topology Abstraction</td>
<td>[WSON-topo]</td>
</tr>
<tr>
<td>Flexi-grid Topology Abstraction</td>
<td>[Flexi-topo]</td>
</tr>
<tr>
<td>Microwave Topology Abstraction</td>
<td>[MW-topo]</td>
</tr>
<tr>
<td>OTN Tunnel Model</td>
<td>[OTN-Tunnel]</td>
</tr>
<tr>
<td>WSON TE Tunnel Model</td>
<td>[WSON-Tunnel]</td>
</tr>
<tr>
<td>Flexi-grid Tunnel Model</td>
<td>[Flexigrid-Tunnel]</td>
</tr>
</tbody>
</table>

* This function is a combination of tunnel set up and client signal description. Usually a tunnel is setting up first to get prepared to carry a client signal, in order to do the service provisioning. Then the client signal is adapted to the established tunnel. It is worth noting that various tunnel models such as [OTN-Tunnel] and [WSON-Tunnel] can be used together with the [TE-tunnel] model to construct technology-specific tunnels, and carry different types of client signals. More details can be found in [Client-signal].

[TE-topo-tunnel] provides the clarification and example usage for TE topology model [TE-topology] and TE tunnel model [TE-tunnel]. [T-NBI Applicability] provides a summary on the applicability of existing YANG model usage in the current network configuration, especially for transport network.

4.4. Device Models in ACTN Architecture (SBI)

Note that SBI is not in the scope of ACTN, as there is already mature protocol solutions for various purpose on the device level of ACTN architecture, such as RSVP-TE, OSPF-TE and so on. The interworking of such protocols and ACTN controller hierarchies can be found in [gmpls-controller-inter-work].
For the device YANG models are used for per-device configuration purpose, they can be used between the PNC and the physical network/devices. One example of Device Models is ietf-te-device yang module defined in [TE-tunnel].

5. Examples of Using Different Types of YANG Models

This section provides some examples on the usage of IETF YANG models in the network operation. A few typical generic scenarios are involved. In [T-NBI Applicability], there are more transport-related scenarios and examples.

5.1. Topology Collection

Before any connection is requested and delivered, the controller needs to understand the network topology. The topology information is exchanged among controllers with topology models, such as [TE-topology]. Moreover, technology-specific topology reporting may use the model described in [OTN-topo] [WSON-topo], and [Flexi-topo] for OTN, WSON and Flexi-grid, respectively. By collecting the network topology, each controller can therefore construct a local database, which can be used for the further service deployment.

There can be different types of abstraction applied between each pair of controllers, corresponding method can be found in [RFC8453]. The technology-specific features may be hidden after abstraction, to make the network easier for the user to operate.

When there is a topology change in the physical network, the PNC should report the change to upper level of controllers via updating messages using topology models. Accordingly, such changes is propagated between different controllers for further synchronization.

5.2. Connectivity over Two Nodes

The service models, such as described in [RFC8299], [RFC8466] and [LiCSM] provide a customer service model which can be used in provider networks.

It would be used as follows in the ACTN architecture:

A CNC uses the service models to specify the two client nodes that are to be connected, and also indicates the amount of traffic (i.e., the bandwidth required) and payload type. What
may be additionally specified is the SLA that describes the required quality and resilience of the service.

The MDSC uses the information in the request to pick the right network (domain) and also to select the provider edge nodes corresponding to the customer edge nodes.

If there are multiple domains, then the MDSC needs to coordinate across domains to set up network tunnels to deliver a service. Thus coordination includes, but is not limited to, picking the right domain sequence to deliver a service.

Additionally, an MDSC can initiate the creation of a tunnel (or tunnel segment) in order to fulfill the service request from CNC based on path computation upon the overall topology information it synthesized from different PNCs. The based model that can cater this purpose is the TE tunnel model specified in [TE-tunnel]. Technology-specific tunnel configuration may use the model described in [OTN-Tunnel], [WSON-Tunnel], and [Flexigrid-Tunnel] for OTN, WSON and Flexi-grid, respectively.

Then, the PNCs need to decide the explicit route of such a tunnel or tunnel segment (in case of multiple domains) for each domain, and then create such a tunnel using protocols such as PCEP and RSVP-TE or using per-hop configuration.

5.3. VN service example

The service model defined in [ACTN-VN-YANG] describes a virtual network (VN) as a service which is a set of multiple connectivity services:

A CNC will request VN to the MDSC by specifying a list of VN members. Each VN member specifies either a single connectivity service, or a source with multiple potential destination points in the case that the precise destination sites are to be determined by MDSC.

- In the first case, the procedure is the same as the connectivity service, except that in this case, there is a list of connections requested.

- In the second case, where the CNC requests the MDSC to select the right destination out of a list of candidates, the MDSC needs to evaluate each candidate and then choose the best one and reply with the chosen destination for a given VN member. After this is selected, the connectivity
request setup procedure is the same as in the connectivity example in section 5.2.

After the VN is set up, a successful reply message is sent from MDSC to CNC, indicating the VN is ready. This message can also be achieved by using the model defined in [ACTN-VN-YANG].

5.4. Data Center-Interconnection Example

This section describes more concretely how existing YANG models described in Section 4 map to an ACTN data center interconnection use case. Figure 3 shows a use-case which shows service policy-driven Data Center selection and is a reproduction of Figure A.1 from [RFC8454].
Figure 3: Service Policy-driven Data Center Selection
Figure 3 shows how VN policies from the CNC (Global Data Center Operation) are incorporated by the MDSC to support multi-destination applications. Multi-destination applications refer to applications in which the selection of the destination of a network path for a given source needs to be decided dynamically to support such applications.

Data Center selection problems arise for VM mobility, disaster recovery and load balancing cases. VN's policy plays an important role for virtual network operation. Policy can be static or dynamic. Dynamic policy for data center selection may be placed as a result of utilization of data center resources supporting VMs. The MDSC would then incorporate this information to meet the objective of this application.

5.4.1. CMI (CNC-MDSC Interface)

[ACTN-VN-YANG] is used to express the definition of a VN, its VN creation request, the service objectives (metrics, QoS parameters, etc.), dynamic service policy when VM needs to be moved from one Data Center to another Data Center, etc. This service model is used between the CNC and the MDSC (CMI). The CNC in this use-case is an external entity that wants to create a VN and operates on the VN.

5.4.2. MPI (MDSC-PNC Interface)

The Network Configuration Model is used between the MDSC and the PNCs. Based on the Customer Service Model’s request, the MDSC will need to translate the service model into the network configuration model to instantiate a set of multi-domain connections between the prescribed sources and the destinations. The MDSC will also need to dynamically interact with the CNC for dynamic policy changes initiated by the CNC. Upon the determination of the multi-domain connections, the MDSC will need to use the network configuration model such as [TE-tunnel] to interact with each PNC involved on the path. [TE-topology] is used to for the purpose of underlying domain network abstraction from the PNC to the MDSC.

5.4.3. SBI (Southbound interface between PNC and devices)

The Device Model can be used between the PNC and its underlying devices that are controlled by the PNC. The PNC will need to trigger signaling using any mechanisms it employs (e.g. [RSVP-TE-YANG]) to provision its domain path segment. There can be a plethora of choices how to control/manage its domain network. The PNC is responsible to abstract its domain network resources and update it.
to the MDSC. Note that this interface is not in the scope of ACTN. This section is provided just for an illustration purpose.

6. Security

This document is an informational draft. When the models mentioned in this draft are implemented, detailed security consideration will be given in such work.

How security fits into the whole architecture has the following components:

- the use of RESTCONF security between components

- the use of authentication and policy to govern which services can be requested by different parties.

- how security may be requested as an element of a service and mapped down to protocol security mechanisms as well as separation (slicing) of physical resources

7. Acknowledgements

We thank Adrian Farrel for providing useful comments and suggestions for this draft.

8. References

8.1. Informative References


9. Contributors

Contributor’s Addresses

Lee, et al.             Expires August 23, 2019
Dhruv Dhody  
Huawei Technologies  
Email: dhruv.ietf@gmail.com  

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

Authors’ Addresses

Young Lee  
Huawei Technologies  
5340 Legacy Drive  
Plano, TX 75023, USA  
Phone: (469)277-5838  
Email: leeyoung@huawei.com

Haomian Zheng  
Huawei Technologies  
Email: zhenghaomian@huawei.com

Daniele Ceccarelli  
Ericsson  
Torshamnsgatan,48  
Stockholm, Sweden  
Email: daniele.ceccarelli@ericsson.com

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

Oscar Gonzalez de Dios  
Telefonica
Email: oscar.gonzalezdedios@telefonica.com

Jong Yoon Shin
SKT

Email: jongyoon.shin@sk.com

Sergio Belotti
Nokia

Email: sergio.belotti@nokia.com
A Framework for Enhanced Virtual Private Networks (VPN+) Service
draft-ietf-teas-enhanced-vpn-01

Abstract

This document specifies a framework for using existing, modified and potential new networking technologies as components to provide an Enhanced Virtual Private Networks (VPN+) service. The purpose is to support the needs of new applications, particularly applications that are associated with 5G services by utilizing an approach that is based on existing VPN technologies and adds features that specific services require over and above traditional VPNs.

Typically, VPN+ will be used to form the underpinning of network slicing, but could also be of use in its own right. It is not envisaged that large numbers of VPN+ instances will be deployed in a network and, in particular, it is not intended that all VPNs supported by a network will use VPN+ techniques.

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 cited them other than as "work in progress."

This Internet-Draft will expire on August 18, 2019.
Table of Contents

1. Introduction .......................... 3
2. Overview of the Requirements .......... 5
   2.1. Isolation between Virtual Networks .. 5
       2.1.1. A Pragmatic Approach to Isolation . . 7
   2.2. Performance Guarantee ............... 8
   2.3. Integration ........................ 9
       2.3.1. Abstraction ................... 10
   2.4. Dynamic Configuration ............... 10
   2.5. Customized Control .................. 11
   2.6. Applicability ...................... 11
3. Architecture of Enhanced VPN .......... 11
   3.1. Layered Architecture ............... 13
   3.2. Multi-Point to Multi-Point .......... 14
   3.3. Application Specific Network Types .. 14
   3.4. Scaling Considerations ............. 14
4. Candidate Technologies ................. 15
   4.1. Underlay Packet and Frame-Based Data Planes .. 15
       4.1.1. FlexE ........................ 16
       4.1.2. Dedicated Queues ............... 16
       4.1.3. Time Sensitive Networking ...... 17
   4.2. Packet and Frame-Based Network Layer ...... 17
       4.2.1. Deterministic Networking ....... 17
       4.2.2. MPLS Traffic Engineering (MPLS-TE) .. 18
       4.2.3. Segment Routing ............... 18
   4.3. Non-Packet Technologies ............ 20
   4.4. Control Plane ..................... 20
   4.5. Management Plane ................... 21
   4.6. Applicability of ACTN to Enhanced VPN . . 21
       4.6.1. ACTN Used for VPN+ Delivery .... 23
       4.6.2. Enhanced VPN Features with ACTN .. 25
5. Scalability Considerations ............ 27
1. Introduction

Virtual private networks (VPNs) have served the industry well as a means of providing different groups of users with logically isolated access to a common network. The common or base network that is used to provide the VPNs is often referred to as the underlay, and the VPN is often called an overlay.

Customers of a network operator may request enhanced overlay services with advanced characteristics such as complete isolation from other services so that changes in network load or event of other services have no effect on the throughput or latency of the service provided to the customer.

Driven largely by needs surfacing from 5G, the concept of network slicing has gained traction [NGMN-NS-Concept] [TS23501] [TS28530] [BBF-SD406]. Network slicing requires the underlying network to support partitioning the network resources to provide the client with dedicated (private) networking, computing, and storage resources drawn from a shared pool. The slices may be seen as (and operated as) virtual networks.

Network abstraction is a technique that can be applied to a network domain to select network resources by policy to obtain a view of potential connectivity and a set of service functions.

Network slicing is an approach to network operations that builds on the concept of network abstraction to provide programmability, flexibility, and modularity. It may use techniques such as Software Defined Networking (SDN) [RFC7149] and Network Function Virtualization (NFV) to create multiple logical (virtual) networks, each tailored for a set of services or a particular tenant that share the same set of requirements, on top of a common network. How the network slices are engineered can be deployment-specific.
Thus, there is a need to create virtual networks with enhanced characteristics. The tenant of such a virtual network can require a degree of isolation and performance that previously could only be satisfied by dedicated networks. Additionally, the tenant may ask for some level of control to their virtual networks, e.g., to customize the service forwarding paths in a network slice.

These enhanced properties cannot be met with pure overlay networks, as they require tighter coordination and integration between the underlay and the overlay network. This document introduces a new network service called Enhanced VPN: VPN+. VPN+ refers to a virtual network which has dedicated network resources, including invoked service functions, allocated from the underlay network. Unlike a traditional VPN, an enhanced VPN can achieve greater isolation with strict guaranteed performance. These new properties, which have general applicability, may also be of interest as part of a network slicing solution, but it is not envisaged that VPN+ techniques will be applied to normal VPN services that can continue to be deployed using pre-existing mechanisms. Furthermore, it is not intended that large numbers of VPN+ instances will be deployed within a single network. See Section 5 for a discussion of scalability considerations.

This document specifies a framework for using existing, modified and potential new networking technologies as components to provide a VPN+ service. Specifically we are concerned with:

- The design of the enhanced data plane.
- The necessary protocols in both underlay and the overlay of enhanced VPN.
- The mechanisms to achieve integration between overlay and underlay.
- The necessary Operation, Administration and Management (OAM) methods to instrument an enhanced VPN to make sure that the required Service Level Agreement (SLA) are met, and to take any corrective action to avoid SLA violation, such as switching to an alternate path.

The required network layered structure to achieve this is shown in Section 3.1.

Note that, in this document, the four terms "VPN", "Enhanced VPN" (or "VPN+'), "Virtual Network (VN)", and "Network Slice" may be considered as describing similar concepts dependent on the viewpoint from which they are used.
An enhanced VPN is clearly a form of VPN, but with additional service-specific commitments. Thus, care must be taken with the term "VPN" to distinguish normal or legacy VPNs from VPN+ instances.

A VN is a type of service that connects customer edge points with the additional possibility of requesting further service characteristics in the manner of an enhanced VPN.

An enhanced VPN or VN is made by creating a slice through the resources of the underlay network.

The general concept of network slicing in a TE network is a larger problem space than is addressed by VPN+ or VN, but those concepts are tools to address some aspects or realizations of network slicing.

2. Overview of the Requirements

In this section we provide an overview of the requirements of an enhanced VPN.

2.1. Isolation between Virtual Networks

One element of the SLA demanded for an enhanced VPN is the degree of isolation from other services in the network. Isolation is a feature requested by some particular customers in the network. Such feature is offered by a network operator where the traffic from one service instance is isolated from the traffic of other services. There are different grades of isolation that range from simple separation of traffic on delivery (ensuring that traffic is not delivered to the wrong customer) all the way to complete separation within the underlay so that the traffic from different services use distinct network resources.

The terms hard and soft isolation are introduced to give example of different isolation cases. A VPN has soft isolation if the traffic of one VPN cannot be received by the customers of another VPN. Both IP and MPLS VPNs are examples of soft isolated VPNs because the network delivers the traffic only to the required VPN endpoints. However, with soft isolation, traffic from one or more VPNs and regular network traffic may congest the network resulting in packet loss and delay for other VPNs operating normally. The ability for a VPN to be sheltered from this effect is called hard isolation, and this property is required by some critical applications.

The requirement is for an operator to provide both hard and soft isolation between the tenants/applications using one enhanced VPN and
the tenants/applications using another enhanced VPN. Hard isolation is needed so that applications with exacting requirements can function correctly, despite other demands (perhaps a burst on another VPN) competing for the underlying resources. In practice isolation may be offered as a spectrum between soft and hard, and in some cases soft and hard isolation may be used in a hierarchical manner.

An example of hard isolation is a network supporting both emergency services and public broadband multi-media services. During a major incident the VPNs supporting these services would both be expected to experience high data volumes, and it is important that both make progress in the transmission of their data. In these circumstances the VPNs would require an appropriate degree of isolation to be able to continue to operate acceptably.

In order to provide the required isolation, resources may have to be reserved in the data plane of the underlay network and dedicated to traffic from a specific VPN. This may introduce scalability concerns, thus some trade-off needs to be considered to provide the required isolation between network slices while still allowing reasonable sharing inside each network slice.

An optical layer can offer a high degree of isolation, at the cost of allocating resources on a long term and end-to-end basis. Such an arrangement means that the full cost of the resources must be borne by the service that is allocated with the resources. On the other hand, where adequate isolation can be achieved at the packet layer, this permits the resources to be shared amongst many services and only dedicated to a service on a temporary basis. This in turn, allows greater statistical multiplexing of network resources and thus amortizes the cost over many services, leading to better economy. However, the degree of isolation required by network slicing cannot be entirely met with existing mechanisms such as Traffic Engineered Label Switched Paths (TE-LSPs). This is because most implementations enforce the bandwidth in the data-plane only at the PEs, but at the P routers the bandwidth is only reserved in the control plane, thus bursts of data can accidentally occur at a P router with higher than committed data rate.

There are several new technologies that provide some assistance with these data plane issues. Firstly there is the IEEE project on Time Sensitive Networking [TSN] which introduces the concept of packet scheduling of delay and loss sensitive packets. Then there is [FLEXE] which provides the ability to multiplex multiple channels over one or more Ethernet links in a way that provides hard isolation. Finally there are advanced queueing approaches which allow the construction of virtual sub-interfaces, each of which is
provided with dedicated resource in a shared physical interface. These approaches are described in more detail later in this document.

In the remainder of this section we explore how isolation may be achieved in packet networks.

2.1.1. A Pragmatic Approach to Isolation

A key question is whether it is possible to achieve hard isolation in packet networks, which were never designed to support hard isolation. On the contrary, they were designed to provide statistical multiplexing, a significant economic advantage when compared to a dedicated, or a Time Division Multiplexing (TDM) network. However there is no need to provide any harder isolation than is required by the application. Pseudowires [RFC3985] emulate services that would have had hard isolation in their native form. An approximation to this requirement is sufficient in most cases.

Thus, for example, using FlexE or a channelized sub-interface together with packet scheduling as interface slicing, optionally along with the slicing of node resources, a type of hard isolation can be provided that is adequate for many VPN+ applications. Other applications may be either satisfied with a classical VPN with or without reserved bandwidth, or may need dedicated point to point fiber. The needs of each application must be quantified in order to provide an economic solution that satisfies those needs without over-engineering.

This spectrum of isolation is shown in Figure 1:

```
O=================================================O
     \------------------------------v-------------------------------/
   Statistical                Pragmatic             Absolute
       Multiplexing               Isolation            Isolation
      (Traditional VPNs)         (Enhanced VPN)       (Dedicated Network)
```

Figure 1: The Spectrum of Isolation

At one end of the above figure, we have traditional statistical multiplexing technologies that support VPNs. This is a service type that has served the industry well and will continue to do so. At the opposite end of the spectrum we have the absolute isolation provided by traditional transport networks. The goal of enhanced VPN is pragmatic isolation. This is isolation that is better than is obtainable from pure statistical multiplexing, more cost effective and flexible than a dedicated network, but which is a practical solution that is good enough for the majority of applications.
2.2. Performance Guarantee

There are several kinds of performance guarantees, including guaranteed maximum packet loss, guaranteed maximum delay and guaranteed delay variation. Note that these guarantees apply to the conformance traffic, the out-of-profile traffic will be handled following other requirements.

Guaranteed maximum packet loss is a common parameter, and is usually addressed by setting the packet priorities, queue size and discard policy. However this becomes more difficult when the requirement is combined with the latency requirement. The limiting case is zero congestion loss, and that is the goal of the Deterministic Networking work that the IETF [DETNET] and IEEE [TSN] are pursuing. In modern optical networks, loss due to transmission errors is already approaches zero, but there are the possibilities of failure of the interface or the fiber itself. This can only be addressed by some form of signal duplication and transmission over diverse paths.

Guaranteed maximum latency is required in a number of applications particularly real-time control applications and some types of virtual reality applications. The work of the IETF Deterministic Networking (DetNet) Working Group [DETNET] is relevant; however the scope needs to be extended to methods of enhancing the underlay to better support the delay guarantee, and to integrate these enhancements with the overall service provision.

Guaranteed maximum delay variation is a service that may also be needed. [I-D.ietf-detnet-use-cases] calls up a number of cases where this is needed, for example electrical utilities have an operational need for this. Time transfer is one example of a service that needs this, although it is in the nature of time that the service might be delivered by the underlay as a shared service and not provided through different virtual networks. Alternatively a dedicated virtual network may be used to provide this as a shared service.

This suggests that a spectrum of service guarantee be considered when deploying an enhanced VPN. As a guide to understanding the design requirements we can consider four types:

- Best effort
- Assured bandwidth
- Guaranteed latency
- Enhanced delivery
Best effort service is the basic service that current VPNs can provide.

An assured bandwidth service is one in which the bandwidth over some period of time is assured, this can be achieved either simply based on best effort with over-capacity provisioning, or it can be based on TE-LSPs with bandwidth reservation. The instantaneous bandwidth is however, not necessarily assured, depending on the technique used. Providing assured bandwidth to VPNs, for example by using TE-LSPs, is not widely deployed at least partially due to scalability concerns.

Guaranteed latency and enhanced delivery are not yet integrated with VPNs.

A guaranteed latency service has a latency upper bound provided by the network. Assuring the upper bound is more important than achieving the minimum latency.

In Section 2.1 we considered the work of the IEEE Time Sensitive Networking (TSN) project [TSN] and the work of the IETF DetNet Working group [DETNET] in the context of isolation. The TSN and DetNet work is of greater relevance in assuring end-to-end packet latency. It is also of importance in considering enhanced delivery.

An enhanced delivery service is one in which the underlay network (at layer 3) attempts to deliver the packet through multiple paths in the hope of eliminating packet loss due to equipment or media failures.

It is these last two characteristics that an enhanced VPN adds to a VPN service.

Flex Ethernet [FLEXE] is a useful underlay to provide these guarantees. This is a method of providing time-slot based channelization over an Ethernet bearer. Such channels are fully isolated from other channels running over the same Ethernet bearer. As noted elsewhere this produces hard isolation but makes the reclamation of unused bandwidth more difficult.

These approaches can be used in tandem. It is possible to use FlexE to provide tenant isolation, and then to use the TSN/Detnet approach to provide a performance guarantee inside the a slice or tenant VPN.

2.3. Integration

A solution to the enhanced VPN problem has to provide close integration of both overlay VPN and the underlay network resource. This needs be done in a flexible and scalable way so that it can be widely deployed in operator networks to support a reasonable number of enhanced VPN customers.
Taking mobile networks and in particular 5G into consideration, the integration of network and the service functions is a likely requirement. The work in IETF SFC working group [SFC] provides a foundation for this integration.

2.3.1. Abstraction

Integration of the overlay VPN and the underlay network resources does not need to be a tight mapping. As described in [RFC7926], abstraction is the process of applying policy to a set of information about a TE network to produce selective information that represents the potential ability to connect across the network. The process of abstraction presents the connectivity graph in a way that is independent of the underlying network technologies, capabilities, and topology so that the graph can be used to plan and deliver network services in a uniform way.

Virtual networks can be built on top of an abstracted topology that represents the connectivity capabilities of the underlay network as described in the framework for Abstraction and Control of TE Networks (ACTN) described in [RFC8453] as discussed further in Section 4.5.

2.4. Dynamic Configuration

Enhanced VPNs need to be created, modified, and removed from the network according to service demand. An enhanced VPN that requires hard isolation must not be disrupted by the instantiation or modification of another enhanced VPN. Determining whether modification of an enhanced VPN can be disruptive to that VPN, and in particular the traffic in flight will be disrupted can be a difficult problem.

The data plane aspects of this problem are discussed further in Section 4.

The control plane aspects of this problem are discussed further in Section 4.4.

The management plane aspects of this problem are discussed further in Section 4.5.

Dynamic changes both to the VPN and to the underlay transport network need to be managed to avoid disruption to sensitive services.

In addition to non-disruptively managing the network as a result of gross change such as the inclusion of a new VPN endpoint or a change to a link, VPN traffic might need to be moved as a result of traffic volume changes.
2.5. Customized Control

In some cases it is desirable that an enhanced VPN has a customized control plane, so that the tenant of the enhanced VPN can have some control to the resources and functions allocated to this enhanced VPN. For example, the tenant may be able to specify the service paths in his own enhanced VPN. Depending on the requirement, an enhanced VPN may have its own dedicated controller, or it may be provided with an interface to a control system which is shared with a set of other tenants, or it may be provided with an interface to the control system provided by the network operator.

Further detail on this requirement will be provided in a future version of the draft. A description of the management plane aspects of this feature can be found in Section 4.5.

2.6. Applicability

The technologies described in this document should be applicable to a number types of VPN services such as:

- Layer 2 point to point services such as pseudowires [RFC3985]
- Layer 2 VPNs [RFC4664]
- Ethernet VPNs [RFC7209]
- Layer 3 VPNs [RFC4364], [RFC2764]
- Virtual Networks (VNs) [RFC8453]

Where such VPN or VN types need enhanced isolation and delivery characteristics, the technology described here can be used to provide an underlay with the required enhanced performance.

3. Architecture of Enhanced VPN

A number of enhanced VPN services will typically be provided by a common network infrastructure. Each enhanced VPN consists of both the overlay and a specific set of dedicated network resources and functions allocated in the underlay to satisfy the needs of the VPN tenant. The integration between overlay and various underlay resources ensures the isolation between different enhanced VPNs, and achieves the guaranteed performance for different services.

An enhanced VPN needs to be designed with consideration given to:

- A enhanced data plane
A control plane to create enhanced VPN, making use of the data plane isolation and guarantee techniques

A management plane for enhanced VPN service life-cycle management

These required characteristics are expanded below:

- Enhanced data plane
  - Provides the required resource isolation capability, e.g. bandwidth guarantee.
  - Provides the required packet latency and jitter characteristics
  - Provides the required packet loss characteristics
  - Provides the mechanism to identify network slice and the associated resources

- Control plane
  - Collect the underlying network topology and resources available and export this to other nodes and/or the centralized controller as required.
  - Create the required virtual networks with the resource and properties needed by the enhanced VPN services that are assigned to it.
  - Determine the risk of SLA violation and take appropriate avoiding action
  - Determine the right balance of per-packet and per-node state according to the needs of enhanced VPN service to scale to the required size

- Management plane
  - Provides the life-cycle management (creation, modification, decommissioning) of enhanced VPN
  - Provides an interface between the enhanced VPN provider and the enhanced VPN clients such that some of the operation requests can be met without interfering with the enhanced VPN of other clients.
3.1. Layered Architecture

The layered architecture of enhanced VPN is shown in Figure 2.

Underpinning everything is the physical infrastructure layer consisting of partitioned links and nodes which provide the underlying resources used to provision the separated virtual networks. Various components and techniques as discussed in Section 4 can be used to provide the resource partition, such as FlexE, Time Sensitive Networking, Deterministic Networking, etc. These partitions may be physical, or virtual so long as the SLA required by the higher layers is met.

These techniques can be used to provision the virtual networks with dedicated resources that they need. To get the required
functionality there needs to be integration between these overlays and the underlay providing the physical resources.

The centralized controller is used to create the virtual networks, to allocate the resources to each virtual network and to provision the enhanced VPN services within the virtual networks. A distributed control plane may also be used for the distribution of the topology and attribute information of the virtual networks.

The creation and allocation process needs to take a holistic view of the needs of all of its tenants, and to partition the resources accordingly. However within a virtual network these resources can if required be managed via a dynamic control plane. This provides the required scalability and isolation.

3.2. Multi-Point to Multi-Point

At the VPN service level, the connectivity are usually mesh or partial-mesh. To support such kind of VPN service, the corresponding underlay is also an abstract MP2MP medium. However when service guarantees are provided, the point-to-point path through the underlay of the enhanced VPN needs to be specifically engineered to meet the required performance guarantees.

3.3. Application Specific Network Types

Although a lot of the traffic that will be carried over the enhanced VPN will likely be IPv4 or IPv6, the design has to be capable of carrying other traffic types, in particular Ethernet traffic. This is easily accomplished through the various pseudowire (PW) techniques [RFC3985]. Where the underlay is MPLS, Ethernet can be carried over the enhanced VPN encapsulated according to the method specified in [RFC4448]. Where the underlay is IP, Layer Two Tunneling Protocol - Version 3 (L2TPv3) [RFC3931] can be used with Ethernet traffic carried according to [RFC4719]. Encapsulations have been defined for most of the common layer two type for both PW over MPLS and for L2TPv3.

3.4. Scaling Considerations

VPNs are instantiated as overlays on top of an operators network and offered as services to the operators customers. An important feature of overlays is that they are able to deliver services without placing per-service state in the core of the underlay network.

Enhanced VPNs may need to install some additional state within the network to achieve the additional features that they require. Solutions must consider minimising and controlling the scale of such
state, and deployment architectures should constrain the number of enhanced VPNs that would exist where such services would place additional state in the network. It is expected that the number of enhanced VPN would be a small number in the beginning, and even in future the number of enhanced VPN will be much less than traditional VPNs, because traditional VPN would be enough for most existing services.

In general, it is not required that the state in the network to be maintained in a 1:1 relationship with the VPN+ instances. It will usually be possible to aggregate a set of VPN+ services so that they share a same set of network resources (much in the way that current VPNs are aggregated over transport tunnels) so that collections of enhanced VPNs that require the same behaviour from the network in terms of resource reservation, latency bounds, resiliency, etc. are able to be grouped together. This is an important feature to assist with the scaling characteristics of VPN+ deployments.

See Section 5 for a greater discussion of scalability considerations.

4. Candidate Technologies

A VPN is a network created by applying a multiplexing technique to the underlying network (the underlay) in order to distinguish the traffic of one VPN from that of another. A VPN path that travels by other than the shortest path through the underlay normally requires state in the underlay to specify that path. State is normally applied to the underlay through the use of the RSVP Signaling protocol, or directly through the use of an SDN controller, although other techniques may emerge as this problem is studied. This state gets harder to manage as the number of VPN paths increases. Furthermore, as we increase the coupling between the underlay and the overlay to support the enhanced VPN service, this state will increase further.

In an enhanced VPN different subsets of the underlay resources are dedicated to different enhanced VPNs. Any enhanced VPN solution thus needs tighter coupling with underlay than is the case with existing VPNs. We cannot for example share the tunnel between enhanced VPNs which require hard isolation.

4.1. Underlay Packet and Frame-Based Data Planes

A number of candidate underlay packet or frame-based data plane solutions which can be used provide the required isolation and guarantee are described in following sections.

- FlexE
4.1.1. FlexE

FlexE [FLEXE] is a method of creating a point-to-point Ethernet with a specific fixed bandwidth. FlexE provides the ability to multiplex multiple channels over an Ethernet link in a way that provides hard isolation. FlexE also supports the bonding of multiple links, which can be used to create larger links out of multiple slower links in a more efficient way than traditional link aggregation. FlexE also supports the sub-rating of links, which allows an operator to only use a portion of a link. However, it is only a link-level technology. When packets are received by the downstream node, they need to be processed in a way that preserves that isolation in the downstream node. This in turn requires a queuing and forwarding implementation that preserves the end-to-end isolation.

If different FlexE channels are used for different services, then no sharing is possible between the FlexE channels. This in turn means that it may be difficult to dynamically redistribute unused bandwidth to lower priority services. This may increase the cost of providing services on the network. On the other hand, FlexE can be used to provide hard isolation between different tenants on a shared interface. The tenant can then use other methods to manage the relative priority of their own traffic in each FlexE channel.

Methods of dynamically re-sizing FlexE channels and the implication for enhanced VPN is for further study.

4.1.2. Dedicated Queues

In order to provide multiple isolated virtual networks for enhanced VPN, the conventional Diff-Serv based queuing system [RFC2475] [RFC4594] is insufficient, due to the limited number of queues which cannot differentiate between traffic of different enhanced VPNs, and the range of service classes that each need to provide to their tenants. This problem is particularly acute with an MPLS underlay due to the small number of traffic class services available. In order to address this problem and reduce the interference between enhanced VPNs, it is necessary to steer traffic of VPNs to dedicated input and output queues. Routers usually have large amounts of queues and sophisticated queuing systems, which could be used or enhanced to provide the levels of isolation required by the applications of enhanced VPN. For example, on one physical interface, the queuing system can provide a set of virtual sub-interfaces, each allocated with dedicated queueing and buffer resources. Sophisticated queuing
systems of this type may be used to provide end-to-end virtual isolation between traffic of different enhanced VPNs.

4.1.3. Time Sensitive Networking

Time Sensitive Networking (TSN) [TSN] is an IEEE project that is designing a method of carrying time sensitive information over Ethernet. It introduces the concept of packet scheduling where a high priority packet stream may be given a scheduled time slot thereby guaranteeing that it experiences no queuing delay and hence a reduced latency. However, when no scheduled packet arrives, its reserved time-slot is handed over to best effort traffic, thereby improving the economics of the network. The mechanisms defined in TSN can be used to meet the requirements of time sensitive services of an enhanced VPN.

Ethernet can be emulated over a Layer 3 network using a pseudowire. However the TSN payload would be opaque to the underlay and thus not treated specifically as time sensitive data. The preferred method of carrying TSN over a layer 3 network is through the use of deterministic networking as explained in the following section of this document.

4.2. Packet and Frame-Based Network Layer

We now consider the problem of slice differentiation and resource representation in the overlay network. The candidate technologies are:

- Deterministic Networking
- MPLS-TE
- Segment Routing

4.2.1. Deterministic Networking

Deterministic Networking (DetNet) [I-D.ietf-detnet-architecture] is a technique being developed in the IETF to enhance the ability of layer 3 networks to deliver packets more reliably and with greater control over the delay. The design cannot use re-transmission techniques such as TCP since that can exceed the delay tolerated by the applications. Even the delay improvements that are achieved with Stream Control Transmission Protocol Partial Reliability Extension (SCTP-PR) [RFC3758] do not meet the bounds set by application demands. DetNet pre-emptively sends copies of the packet over various paths to minimize the chance of all packets being lost, and trims duplicate packets to prevent excessive flooding of the network.
and to prevent multiple packets being delivered to the destination. It also seeks to set an upper bound on latency. The goal is not to minimize latency; the optimum upper bound paths may not be the minimum latency paths.

DetNet is based on flows. It currently does not specify the use of underlay topology other than the base topology. To be of use for enhanced VPN, DetNet needs to be integrated with different virtual topologies of enhanced VPNs.

The detailed design that allows the use DetNet in a multi-tenant network, and how to improve the scalability of DetNet in a multi-tenant network are topics for further study.

4.2.2. MPLS Traffic Engineering (MPLS-TE)

MPLS-TE introduces the concept of reserving end-to-end bandwidth for a TE-LSP, which can be used as the underlay of VPNs. It also introduces the concept of non-shortest path routing through the use of the Explicit Route Object [RFC3209]. VPN traffic can be run over dedicated TE-LSPs to provide reserved bandwidth for each specific connection in a VPN. Some network operators have concerns about the scalability and management overhead of RSVP-TE system, and this has lead them to consider other solutions for their networks.

4.2.3. Segment Routing

Segment Routing [RFC8402] is a method that prepends instructions to packets at the head-end node and optionally at various points as it passes through the network. These instructions allow the packets to be routed on paths other than the shortest path for various traffic engineering reasons. These paths can be strict or loose paths, depending on the compactness required of the instruction list and the degree of autonomy granted to the network, for example to support Equal Cost Multipath load-balancing (ECMP) [RFC2992].

With SR, a path needs to be dynamically created through a set of segments by simply specifying the Segment Identifiers (SIDs), i.e. instructions rooted at a particular point in the network. Thus if a path is to be provisioned from some ingress point A to some egress point B in the underlay, A is provided with a SID list from A to B and instructions on how to identify the packets to which the SID list is to be prepended.

By encoding the state in the packet, as is done in Segment Routing, per-path state is transitioned out of the network.
However, there are a number of limitations in current SR, which limit its applicability to enhanced VPNs:

- Segments are shared between different VPNs paths
- There is no reservation of bandwidth
- There is limited differentiation in the data plane.

Thus some extensions to SR are needed to provide isolation between different enhanced VPNs. This can be achieved by including a finer granularity of state in the network in anticipation of its future use by authorized services. We therefore need to evaluate the balance between this additional state and the performance delivered by the network.

With current segment routing, the instructions are used to specify the nodes and links to be traversed. However, in order to achieve the required isolation between different services, new instructions can be created which can be prepended to a packet to steer it through specific network resources and functions.

Traditionally an SR traffic engineered path operates with a granularity of a link with hints about priority provided through the use of the traffic class (TC) field in the header. However to achieve the latency and isolation characteristics that are sought by the enhanced VPN users, steering packets through specific queues and resources will likely be required. The extent to which these needs can be satisfied through existing QoS mechanisms is to be determined. What is clear is that a fine control of which services wait for which, with a fine granularity of queue management policy is needed. Note that the concept of a queue is a useful abstraction for many types of underlay mechanism that may be used to provide enhanced isolation and latency support.

From the perspective of the control plane, and from the perspective of the segment routing, the method of steering a packet to a queue that provides the required properties is an abstraction that hides the details of the underlying implementation. How the queue satisfies the requirement is implementation specific and is transparent to the control plane and data plane mechanisms used. Thus, for example, a FlexE channel, or a time sensitive networking packet scheduling slot are abstracted to the same concept and bound to the data plane in a common manner.

We can also introduce such fine grained packet steering by specifying the queues through an SR instruction list. Thus new SR instructions may be created to specify not only which resources are traversed, but
in some cases how they are traversed. For example, it may be possible to specify not only the queue to be used but the policy to be applied when enqueuing and dequeuing.

This concept could be further generalized, since as well as queuing to the output port of a router, it is possible to consider queuing data to any resource, for example:

- A network processor unit (NPU)
- A central processing unit (CPU) Core
- A Look-up engine

Both SR-MPLS and SRv6 are candidate network layer technologies for enhanced VPN. In some cases they can be supported by DetNet to meet the packet loss, delay and jitter requirement of particular service. However, currently the "pure" IP variant of DetNet [I-D.ietf-detnet-dp-sol-ip] does not support the Packet Replication, Elimination, and Re-ordering (PREOF) [I-D.ietf-detnet-architecture] functions. How to provide the DetNet enhanced delivery in an SRv6 environment needs further study.

4.3. Non-Packet Technologies

Non-packet underlay data plane technologies often have TE properties and behaviors, and meet many of the key requirements in particular for bandwidth guarantees, traffic isolation (with physical isolation often being an integral part of the technology), highly predictable latency and jitter characteristics, measurable loss characteristics, and ease of identification of flows (and hence slices).

The control and management planes for non-packet data plane technologies have most in common with MPLS-TE (Section 4.2.2) and offer the same set of advanced features [RFC3945]. Furthermore, management techniques such as ACTN ([RFC8453] and Section 4.4) can be used to aid in the reporting of underlying network topologies, and the creation of virtual networks with the resource and properties needed by the enhanced VPN services.

4.4. Control Plane

Enhanced VPN would likely be based on a hybrid control mechanism, which takes advantage of the logically centralized controller for on-demand provisioning and global optimization, whilst still relies on distributed control plane to provide scalability, high reliability, fast reaction, automatic failure recovery etc. Extension and
optimization to the distributed control plane is needed to support the enhanced properties of VPN+.

RSVP-TE provides the signaling mechanism of establishing a TE-LSP with end-to-end resource reservation. It can be used to bind the VPN to specific network resource allocated within the underlay, but there are the above mentioned scalability concerns.

SR does not have the capability of signaling the resource reservation along the path, nor do its currently specified distributed link state routing protocols. On the other hand, the SR approach provides a way of efficiently binding the network underlay and the enhanced VPN overlay, as it reduces the amount of state to be maintained in the network. An SR-based approach with per-slice resource reservation can easily create dedicated SR network slices, and the VPN services can be bound to a particular SR network slice. A centralized controller can perform resource planning and reservation from the controller's point of view, but this does not ensure resource reservation is actually done in the network nodes. Thus, if a distributed control plane is needed, either in place of an SDN controller or as an assistant to it, the design of the control system needs to ensure that resources are uniquely allocated in the network nodes for the correct service, and not allocated to multiple services causing unintended resource conflict.

4.5. Management Plane

The management plane mechanisms for enhanced VPN can be based on the VPN service models as defined in [RFC8299] and [RFC8466], possible augmentations and extensions to these models may be needed, which is out of the scope of this document.

Abstraction and Control of Traffic Engineered Networks (ACTN) [RFC8453] specifies the SDN based architecture for the control of TE networks. The ACTN related data models such as [I-D.ietf-teas-actn-vn-yang] and [I-D.lee-teas-te-service-mapping-yang] can be applicable in the provisioning of enhanced VPN service. The details are described in Section 4.6.

4.6. Applicability of ACTN to Enhanced VPN

ACTN facilitates end-to-end connections and provides them to the user. The ACTN framework [RFC8453] highlights how:

- Abstraction of the underlying network resources are provided to higher-layer applications and customers.
Virtualization of underlying resources, whose selection criterion is the allocation of those resources for the customer, application, or service.

Creation of a virtualized environment allowing operators to view and control multi-domain networks as a single virtualized network.

The presentation to customers of networks as a virtual network via open and programmable interfaces.

The infrastructure managed through ACTN comprises traffic engineered network resources, which may include:

- Statistical packet bandwidth.
- Physical forwarding plane sources, such as: wavelengths and time slots.
- Forwarding and cross-connect capabilities.

The type of network virtualization enabled by ACTN provides customers and applications (tenants) with the capability to utilize and independently control allocated virtual network resources as if they were physically their own resources.

An ACTN Virtual Network (VN) is a client view of the ACTN managed infrastructure, and is presented by the ACTN provider as a set of abstracted resources.

Depending on the agreement between client and provider various VN operations and VN views are possible.

- Virtual Network Creation: A VN could be pre-configured and created via static or dynamic request and negotiation between customer and provider. It must meet the specified SLA attributes which satisfy the customer’s objectives.

- Virtual Network Operations: The virtual network may be further modified and deleted based on customer request to request changes in the network resources reserved for the customer, and used to construct the network slice. The customer can further act upon the virtual network to manage traffic flow across the virtual network.

- Virtual Network View: The VN topology from a customer point of view. These may be a variety of tunnels, or an entire VN topology. Such connections may comprise of customer end points, access links, intra-domain paths, and inter-domain links.
Dynamic VN Operations allow a customer to modify or delete the VN. The customer can further act upon the virtual network to create/modify/delete virtual links and nodes. These changes will result in subsequent tunnel management in the operator’s networks.

4.6.1. ACTN Used for VPN+ Delivery

ACTN provides VPN connections between multiple sites as requested via a VPN requestor enabled by the Customer Network Controller (CNC). The CNC is managed by the customer themselves, and interacts with the network provider’s Multi-Domain Service Controller (MDSC). The Provisioning Network Controllers (PNC) remain entirely under the management of the network provider and are not visible to the customer.

The benefits of this model include:

- Provision of edge-to-edge VPN multi-access connectivity.
- Management is mostly performed by the network provider, with some flexibility delegated to the customer-managed CNC.
Figure 4 presents a more general representation of how multiple enhanced VPNs may be created from the resources of multiple physical networks using the CNC, MDSC, and PNC components of the ACTN architecture. Each enhanced VPN is controlled by its own CNC. The CNCs send requests to the provider’s MDSC. The provider manages two physical networks each under the control of PNC. The MDSC asks the PNCs to allocate and provision resources to achieve the enhanced VPNs. In this figure, one enhanced VPN is constructed solely from the resources of one of the physical networks, while the other VPN uses resources from both physical networks.
4.6.2. Enhanced VPN Features with ACTN

This section discusses how the features of ACTN can fulfill the enhanced VPN requirements described earlier in this document. As previously noted, key requirements of the enhanced VPN include:

1. Isolation between VPNs
2. Guaranteed Performance
3. Integration
4. Dynamic Configuration
5. Customized Control Plane

The subsections that follow outline how each requirement is met using ACTN.
4.6.2.1. Isolation Between VPNs

The ACTN VN YANG model [I-D.ietf-teas-actn-vn-yang] and the TE-service mapping model [I-D.lee-teas-te-service-mapping-yang] fulfill the VPN isolation requirement by providing the following features for the VNs:

- Each VN is identified with a unique identifier (vn-id and vn-name) and so is each VN member that belongs to the VN (vn-member-id).
- Each instantiated VN is managed and controlled independent of other VNs in the network with proper protection level (protection).
- Each VN is instantiated with an isolation requirement described by the TE-service mapping model [I-D.lee-teas-te-service-mapping-yang]. This mapping supports:
  - Hard isolation with deterministic characteristics (e.g., this case may need an optical bypass tunnel or a DetNet/TSN tunnel to guarantee latency with no jitter)
  - Hard isolation (i.e., dedicated TE resources in all underlays)
  - Soft isolation (i.e., resource in some layer may be shared while in some other layers is dedicated).
  - No isolation (i.e., sharing with other VN).

4.6.2.2. Guaranteed Performance

Performance objectives of a VN need first to be expressed in order to assure the performance guarantee. [I-D.ietf-teas-actn-vn-yang] and [I-D.ietf-teas-yang-te-topo] allow configuration of several parameters that may affect the VN performance objectives as follows:

- Bandwidth
- Objective function (e.g., min cost path, min load path, etc.)
- Metric Types and their threshold:
  - TE cost, IGP cost, Hop count, or Unidirectional Delay (e.g., can set all path delay <= threshold)

Once these requests are instantiated, the resources are committed and guaranteed through the life cycle of the VN.
4.6.2.3. Integration

ACTN provides mechanisms to correlate customer’s VN and the actual TE tunnels instantiated in the provider’s network. Specifically:

- Link each VN member to actual TE tunnel.
- Each VN can be monitored on a various level such as VN level, VN member level, TE-tunnel level, and link/node level.

Service function integration with network topology (L3 and TE topology) is in progress in [I-D.ietf-teas-sf-aware-topo-model]. Specifically, [I-D.ietf-teas-sf-aware-topo-model] addresses a number of use-cases that show how TE topology supports various service functions.

4.6.2.4. Dynamic Configuration

ACTN provides an architecture that allows the CNC to interact with the MDSC which is network provider’s SDN controller. This gives the customer control of their VNs.

Specifically, the ACTN VN model [I-D.ietf-teas-actn-vn-yang] allows the VN to create, modify, and delete VNs.

4.6.2.5. Customized Control

ACTN provides a YANG model that allows the CNC to control a VN as a "Type 2 VN" that allows the customer to provision tunnels that connect their endpoints over the customized VN topology.

For some VN members, the customers are allowed to configure the path (i.e., the sequence of virtual nodes and virtual links) over the VN/abstract topology.

5. Scalability Considerations

Enhanced VPN provides the performance guaranteed services in packet networks, but with the potential cost of introducing additional states into the network. There are at least three ways that this adding state might be presented in the network:

- Introduce the complete state into the packet, as is done in SR. This allows the controller to specify the detailed series of forwarding and processing instructions for the packet as it transits the network. The cost of this is an increase in the packet header size. The cost is also that systems will have capabilities enabled in case they are called upon by a service.
This is a type of latent state, and increases as we more precisely specify the path and resources that need to be exclusively available to a VPN.

- Introduce the state to the network. This is normally done by creating a path using RSVP-TE, which can be extended to introduce any element that needs to be specified along the path, for example explicitly specifying queuing policy. It is of course possible to use other methods to introduce path state, such as via a Software Defined Network (SDN) controller, or possibly by modifying a routing protocol. With this approach there is state per path per path characteristic that needs to be maintained over its life-cycle. This is more state than is needed using SR, but the packet are shorter.

- Provide a hybrid approach based on using binding SIDs to create path fragments, and bind them together with SR.

Dynamic creation of a VPN path using SR requires less state maintenance in the network core at the expense of larger VPN headers on the packet. The packet size can be lower if a form of loose source routing is used (using a few nodal SIDs), and it will be lower if no specific functions or resource on the routers are specified. Reducing the state in the network is important to enhanced VPN, as it requires the overlay to be more closely integrated with the underlay than with traditional VPNs. This tighter coupling would normally mean that more state needed to be created and maintained in the network, as the state about fine granularity processing would need to be loaded and maintained in the routers. However, a segment routed approach allows much of this state to be spread amongst the network ingress nodes, and transiently carried in the packets as SIDs.

These approaches are for further study.

5.1. Maximum Stack Depth of SR

One of the challenges with SR is the stack depth that nodes are able to impose on packets [I-D.ietf-isis-segment-routing-msd]. This leads to a difficult balance between adding state to the network and minimizing stack depth, or minimizing state and increasing the stack depth.

5.2. RSVP Scalability

The traditional method of creating a resource allocated path through an MPLS network is to use the RSVP protocol. However there have been concerns that this requires significant continuous state maintenance
in the network. There are ongoing works to improve the scalability of RSVP-TE LSPs in the control plane [RFC8370].

There is also concern at the scalability of the forwarder footprint of RSVP as the number of paths through an LSR grows [I-D.sitaraman-mpls-rsvp-shared-labels] proposes to address this by employing SR within a tunnel established by RSVP-TE.

6. OAM Considerations

A study of OAM in SR networks has been documented in [RFC8403].

The enhanced VPN OAM design needs to consider the following requirements:

- Instrumentation of the underlay so that the network operator can be sure that the resources committed to a tenant are operating correctly and delivering the required performance.

- Instrumentation of the overlay by the tenant. This is likely to be transparent to the network operator and to use existing methods. Particular consideration needs to be given to the need to verify the isolation and the various committed performance characteristics.

- Instrumentation of the overlay by the network provider to proactively demonstrate that the committed performance is being delivered. This needs to be done in a non-intrusive manner, particularly when the tenant is deploying a performance sensitive application.

- Verification of the conformity of the path to the service requirement. This may need to be done as part of a commissioning test.

These issues will be discussed in a future version of this document.

7. Enhanced Resiliency

Each enhanced VPN has a life-cycle, and needs modification during deployment as the needs of its tenant change. Additionally, as the network as a whole evolves, there will need to be garbage collection performed to consolidate resources into usable quanta.

Systems in which the path is imposed such as SR, or some form of explicit routing tend to do well in these applications, because it is possible to perform an atomic transition from one path to another. This is a single action by the head-end changes the path without the
need for coordinated action by the routers along the path. However, implementations and the monitoring protocols need to make sure that the new path is up and meet the required SLA before traffic is transitioned to it. It is possible for deadlocks arise as a result of the network becoming fragmented over time, such that it is impossible to create a new path or modify an existing path without impacting the SLA of other paths. Resolution of this situation is as much a commercial issue as it is a technical issue and is outside the scope of this document.

There are however two manifestations of the latency problem that are for further study in any of these approaches:

- The problem of packets overtaking one and other if a path latency reduces during a transition.
- The problem of the latency transient in either direction as a path migrates.

There is also the matter of what happens during failure in the underlay infrastructure. Fast reroute is one approach, but that still produces a transient loss with a normal goal of rectifying this within 50ms [RFC5654]. An alternative is some form of N+1 delivery such as has been used for many years to support protection from service disruption. This may be taken to a different level using the techniques proposed by the IETF deterministic network work with multiple in-network replication and the culling of later packets [I-D.ietf-detnet-architecture].

In addition to the approach used to protect high priority packets, consideration has to be given to the impact of best effort traffic on the high priority packets during a transient. Specifically if a conventional re-convergence process is used there will inevitably be micro-loops and whilst some form of explicit routing will protect the high priority traffic, lower priority traffic on best effort shortest paths will micro-loop without the use of a loop prevention technology. To provide the highest quality of service to high priority traffic, either this traffic must be shielded from the micro-loops, or micro-loops must be prevented.

8. Security Considerations

All types of virtual network require special consideration to be given to the isolation between the tenants. In this regard enhanced VPNs neither introduce, no experience a greater security risk than another VPN of the same base type. However, in an enhanced virtual network service the isolation requirement needs to be considered. If a service requires a specific latency then it can be damaged by...
simply delaying the packet through the activities of another tenant. In a network with virtual functions, depriving a function used by another tenant of compute resources can be just as damaging as delaying transmission of a packet in the network. The measures to address these dynamic security risks must be specified as part to the specific solution.

9. IANA Considerations

There are no requested IANA actions.

10. Contributors

Daniel King
Email: daniel@olddog.co.uk

Adrian Farrel
Email: adrian@olddog.co.uk

Jeff Tansura
Email: jefftant.ietf@gmail.com

Qin Wu
Email: bill.wu@huawei.com

Daniele Ceccarelli
Email: daniele.ceccarelli@ericsson.com

Mohamed Boucadair
Email: mohamed.boucadair@orange.com

Sergio Belotti
Email: sergio.belotti@nokia.com

Haomian Zheng
Email: zhenghaomian@huawei.com

11. Acknowledgements

The authors would like to thank Charlie Perkins, James N Guichard and John E Drake for their review and valuable comments.

This work was supported in part by the European Commission funded H2020-ICT-2016-2 METRO-HAUL project (G.A. 761727).
12. References

12.1. Normative References


12.2. Informative References


[I-D.ietf-teas-actn-vn-yang]

[I-D.ietf-teas-sf-aware-topo-model]

[I-D.ietf-teas-yang-te-topo]

[I-D.lee-teas-te-service-mapping-yang]

[I-D.sitaraman-mpls-rsvp-shared-labels]

[NGMN-NS-Concept]


Authors’ Addresses

Jie Dong
Huawei
Email: jie.dong@huawei.com

Stewart Bryant
Huawei
Email: stewart.bryant@gmail.com

Zhenqiang Li
China Mobile
Email: lizhenqiang@chinamobile.com

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

Young Lee  
Huawei  

Email: leeyoung@huawei.com
Traffic Engineering and Service Mapping Yang Model

draft-ietf-teas-te-service-mapping-yang-01

Abstract

This document provides a YANG data model to map customer service models (e.g., the L3VPN Service Model) to Traffic Engineering (TE) models (e.g., the TE Tunnel or the Abstraction and Control of Traffic Engineered Networks Virtual Network model). This model is referred to as TE Service Mapping Model and is applicable generically to the operator's need for seamless control and management of their VPN services with TE tunnel support.

The model is principally used to allow monitoring and diagnostics of the management systems to show how the service requests are mapped onto underlying network resource and TE models.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.
Table of Contents

1. Introduction...................................................3
   1.1. Terminology...............................................4
   1.2. Tree diagram..............................................4
   1.3. Prefixes in Data Node Names...............................4
2. TE & Service Related Parameters................................5
   2.1. VN/Tunnel Selection Requirements..........................5
   2.2. Availability Requirement..................................7
3. YANG Modeling Approach.........................................7
   3.1. Forward Compatibility.....................................8
4. L3VPN Architecture in the ACTN Context.........................8
   4.1. Service Mapping..........................................11
   4.2. Site Mapping.............................................12
5. Applicability of TE-Service Mapping in Generic context........12
6. YANG Data Trees...............................................13
Data models are a representation of objects that can be configured or monitored within a system. Within the IETF, YANG [RFC6020] is the language of choice for documenting data models, and YANG models have been produced to allow configuration or modeling of a variety of network devices, protocol instances, and network services. YANG data models have been classified in [RFC8199] and [RFC8309].

Framework for Abstraction and Control of Traffic Engineered Networks (ACTN) [RFC8453] introduces an architecture to support virtual network services and connectivity services. [ACTN-VN-YANG] defines a YANG model and describes how customers or end-to-end orchestrators can request and/or instantiate a generic virtual network service. [ACTN-Applicability] describes the way IETF YANG models of different classifications can be applied to the ACTN interfaces. In particular, it describes how customer service models can be mapped into the CNC-MDSC Interface (CMI) of the ACTN architecture.

The models presented in this document are also applicable in generic context [RFC8309] as part of Customer Service Model used between Service Orchestrator and Customer.

[RFC8299] provides a L3VPN service delivery YANG model for PE-based VPNs. The scope of that draft is limited to a set of domains under control of the same network operator to deliver services requiring TE tunnels.

[L2SM] provides a L2VPN service delivery YANG model for PE-based VPNs. The scope of that draft is limited to a set of domains under control of the same network operator to deliver services requiring TE tunnels.

[L1CSM] provides a L1 connectivity service delivery YANG model for PE-based VPNs. The scope of that draft is limited to a set of domains under control of the same network operator to deliver services requiring TE tunnels.
While the IP/MPLS Provisioning Network Controller (PNC) is responsible for provisioning the VPN service on the Provider Edge (PE) nodes, the Multi-Domain Service Coordinator (MDSC) can coordinate how to map the VPN services onto Traffic Engineering (TE) tunnels. This is consistent with the two of the core functions of the MDSC specified in [RFC8453]:

- Customer mapping/translation function: This function is to map customer requests/commands into network provisioning requests that can be sent to the PNC according to the business policies that have been provisioned statically or dynamically. Specifically, it provides mapping and translation of a customer’s service request into a set of parameters that are specific to a network type and technology such that the network configuration process is made possible.

- Virtual service coordination function: This function translates customer service-related information into virtual network service operations in order to seamlessly operate virtual networks while meeting a customer’s service requirements. In the context of ACTN, service/virtual service coordination includes a number of service orchestration functions such as multi-destination load balancing, guarantees of service quality, bandwidth and throughput. It also includes notifications for service fault and performance degradation and so forth.

Section 2 describes a set of TE & service related parameters that this document addresses as new and advanced parameters that are not included in generic service models. Section 3 discusses YANG modeling approach.

1.1. Terminology

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

1.2. Tree diagram

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

1.3. Prefixes in Data Node Names

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsm-types</td>
<td>ietf-te-service-mapping-types</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>l1</td>
<td>ietf-l1csm</td>
<td>[L1CSM]</td>
</tr>
<tr>
<td>l2vpn-svc</td>
<td>ietf-l2vpn-svc</td>
<td>[L2SM]</td>
</tr>
<tr>
<td>l3vpn-svc</td>
<td>ietf-l3vpn-svc</td>
<td>[RFC8299]</td>
</tr>
<tr>
<td>l1-tsm</td>
<td>ietf-l1csm-te-service-mapping</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>l2-tsm</td>
<td>ietf-l2sm-te-service-mapping</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>l3-tsm</td>
<td>ietf-l3sm-te-service-mapping</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>vn</td>
<td>ietf-vn</td>
<td>[ACTN-VN]</td>
</tr>
<tr>
<td>nw</td>
<td>ietf-network</td>
<td>[RFC8345]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[TE-Types]</td>
</tr>
<tr>
<td>te-topo</td>
<td>ietf-te-topology</td>
<td>[TE-Topo]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[TE-Tunnel]</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

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

2. TE & Service Related Parameters

While L1/2/3 service models [L1CSM, L2SM, L3SM] are intended to provide service-specific parameters for VPN service instances, there are a number of TE & Service related parameters that are not included in the generic service models.

Additional service parameters and policies that are not included in the aforementioned service models are addressed in the YANG models defined in this document.

2.1. VN/Tunnel Selection Requirements

In some cases, the service requirements may need addition TE tunnels to be established. This may occur when there are no suitable existing TE tunnels that can support the service requirements, or when the operator would like to dynamically create and bind tunnels to the VPN such that they are not shared by other VPNs, for example, for network slicing. The establishment of TE tunnels is subject to the network operator’s policies.
To summarize, there are three modes of VN/Tunnel selection operations to be supported as follows. Additional modes may be defined in the future.

- **New VN/Tunnel Binding** - A customer could request a VPN service based on VN/Tunnels that are not shared with other existing or future services. This might be to meet VPN isolation requirements. Further, the YANG model described in Section 5 of this document can be used to describe the mapping between the VPN service and the ACTN VN. The VN (and TE tunnels) could be bound to the VPN and not used for any other VPN.

  Under this mode, the following sub-categories can be supported:

  1. **Hard Isolation with deterministic characteristics**: A customer could request a VPN service using a set of TE Tunnels with deterministic characteristics requirements (e.g., no latency variation) and where that set of TE Tunnels must not be shared with other VPN services and must not compete for bandwidth or other network resources with other TE Tunnels.

  2. **Hard Isolation**: This is similar to the above case but without the deterministic characteristics requirements.

  3. **Soft Isolation**: The customer requests a VPN service using a set of TE tunnels which can be shared with other VPN services.

- **VN/Tunnel Sharing** - A customer could request a VPN service where new tunnels (or a VN) do not need to be created for each VPN and can be shared across multiple VPNs. Further, the mapping YANG model described in Section 5 of this document can be used to describe the mapping between the VPN service and the tunnels in use. No modification of the properties of a tunnel (or VN) is allowed in this mode: an existing tunnel can only be selected.

- **VN/Tunnel Modify** - This mode allows the modification of the properties of the existing VN/tunnel (e.g., bandwidth).
2.2. Availability Requirement

Availability is another service requirement or intent that may influence the selection or provisioning of TE tunnels or a VN to support the requested service. Availability is a probabilistic measure of the length of time that a VPN/VN instance functions without a network failure.

The availability level will need to be translated into network specific policies such as the protection/reroute policy associated with a VN or Tunnel. The means by which this is achieved is not in the scope of this draft.

3. YANG Modeling Approach

This section provides how the TE & Service mapping parameters are supported using augmentation of the existing service models (i.e., [L1CSM], [L2SM], and [L3SM]). Figure 1 shows the scope of the Augmented LxSM Model.

The Augmented LxSM model (where x=1,2,3) augments the basic LxSM model while importing the common TE & Service related parameters (defined in Section 2) grouping information from TE & Service Mapping Types. The TE & Service Mapping Types (ietf-te-service-mapping-types) module is the repository of all common groupings imported by each augmented LxSM model. Any future service models would import this grouping file.

The role of the augmented LxSm service model is to expose the mapping relationship between service models and TE models so that VN/VPN service instantiations provided by the underlying TE networks
can be viewed outside of the MDSC, for example by an operator who is diagnosing the behavior of the network. It also allows for the customers to access operational state information about how their services are instantiated with the underlying VN, TE topology or TE tunnels provided that the MDSC operator is willing to share that information. This mapping will facilitate a seamless service management operation with underlay-TE network visibility.

As seen in Figure 1, the augmented LxSM service model records a mapping between the customer service models and the ACTN VN YANG model. Thus, when the MDSC receives a service request it creates a VN that meets the customer’s service objectives with various constraints via TE-topology model [TE-topo], and this relationship is recorded by the Augmented LxSM Model. The model also supports a mapping between a service model and TE-topology or a TE-tunnel.

3.1. Forward Compatibility

The YANG module defined in this document supports three existing service models via augmenting while sharing the common TE & Service Mapping Types.

It is possible that new service models will be defined at some future time and that it will be desirable to map them to underlying TE constructs in the same way as the three existing models are augmented.

4. L3VPN Architecture in the ACTN Context

Figure 2 shows the architectural context of this document referencing the ACTN components and interfaces.
Figure 2: L3VPN Architecture from the IP+Optical Network Perspective

There are three main entities in the ACTN architecture and shown in Figure 2.

- CNC: The Customer Network Controller is responsible for generating service requests. In the context of an L3VPN, the CNC uses the Augmented L3SM to express the service request and communicate it to the network operator.
MDSC: This entity is responsible for coordinating a L3VPN service request (expressed via the Augmented L3SM) with the IP/MPLS PNC and the Transport PNC. For TE services, one of the key responsibilities of the MDSC is to coordinate with both the IP PNC and the Transport PNC for the mapping of the Augmented L3VPN Service Model to the ACTN VN model. In the VN/TE-tunnel binding case, the MDSC will need to coordinate with the Transport PNC to dynamically create the TE-tunnels in the transport network as needed. These tunnels are added as links in the IP/MPLS Layer topology. The MDSC coordinates with IP/MPLS PNC to create the TE-tunnels in the IP/MPLS layer, as part of the ACTN VN creation.

PNC: The Provisioning Network Controller is responsible for configuring and operating the network devices. Figure 2 shows two distinct PNCs.

- IP/MPLS PNC (PNC1): This entity is responsible for device configuration to create PE-PE L3VPN tunnels for the VPN customer and for the configuration of the L3VPN VRF on the PE nodes. Each network element would select a tunnel based on the configuration.
- Transport PNC (PNC2): This entity is responsible for device configuration for TE tunnels in the transport networks.

There are four main interfaces shown in Figure 2.

- CMI: The CNC-MDSC Interface is used to communicate service requests from the customer to the operator. The requests may be expressed as Augmented VPN service requests (L2SM, L3SM), as connectivity requests (L1CSM), or as virtual network requests (ACTN VN).
- MPI: The MDSC-PNC Interface is used by the MDSC to orchestrate networks under the control of PNCs. The requests on this interface may use TE tunnel models, TE topology models, VPN network configuration models or layer one connectivity models.
- SBI: The Southbound Interface is used by the PNC to control network devices and is out of scope for this document.
- The TE Service Mapping Model as described in this document can be used to see the mapping between service models and VN models and TE Tunnel/Topology models. That mapping may occur in the CNC if a service request is mapped to a VN request. Or it may occur in the MDSC where a service request is mapped to a TE tunnel, TE topology, or VPN network configuration model. The TE Service Mapping Model may be read from the CNC or MDSC to understand how the mapping has been made and to see the purpose for which network resources are used.
As shown in Figure 2, the MDSC may be used recursively. For example, the CNC might map a L3SM request to a VN request that it sends to a recursive MDSC.

The high-level control flows for one example are as follows:

1. A customer asks for an L3VPN between CE1 and CE2 using the Augmented L3SM model.

2. The MDSC considers the service request and local policy to determine if it needs to create a new VN or any TE Topology, and if that is the case, ACTN VN YANG [ACTN-VN-YANG] is used to configure a new VN based on this VPN and map the VPN service to the ACTN VN. In case an existing tunnel is to be used, each device will select which tunnel to use and populate this mapping information.

3. The MDSC interacts with both the IP/MPLS PNC and the Transport PNC to create a PE-PE tunnel in the IP network mapped to a TE tunnel in the transport network by providing the inter-layer access points and tunnel requirements. The specific service information is passed to the IP/MPLS PNC for the actual VPN configuration and activation.
   a. The Transport PNC creates the corresponding TE tunnel matching with the access point and egress point.
   b. The IP/MPLS PNC maps the VPN ID with the corresponding TE tunnel ID to bind these two IDs.

4. The IP/MPLS PNC creates/updates a VRF instance for this VPN customer. This is not in the scope of this document.

4.1. Service Mapping

Augmented L3SM and L2SM can be used to request VPN service creation including the creation of sites and corresponding site network access connection between CE and PE. A VPN-ID is used to identify each VPN service ordered by the customer. The ACTN VN can be used further to establish PE-to-PE connectivity between VPN sites belonging to the same VPN service. A VN-ID is used to identify each virtual network established between VPN sites.

Once the ACTN VN has been established over the TE network (maybe a new VN, maybe modification of an existing VN, or maybe the use of an unmodified existing VN), the mapping between the VPN service and the ACTN VN service can be created.
4.2. Site Mapping

The elements in Augmented L3SM and L2SM define site location parameters and constraints such as distance and access diversity that can influence the placement of network attachment points (i.e., virtual network access points (VNAP)). To achieve this, a central directory can be set up to establish the mapping between location parameters and constraints and network attachment point location. Suppose multiple attachment points are matched, the management system can use constraints or other local policy to select the best candidate network attachment points.

After a network attachment point is selected, the mapping between VPN site and VNAP can be established as shown in Table 1.

<table>
<thead>
<tr>
<th>Site</th>
<th>Network Access</th>
<th>Location (Address, Postal Code, State, City, Country Code)</th>
<th>Access Diversity (Constraint-Type, Group-id, Target Group-id)</th>
<th>PE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SITE1 ACCESS1</td>
<td>(,,US,NewYork,)</td>
<td>(10,PE-Diverse,10)</td>
<td>PE1</td>
<td></td>
</tr>
<tr>
<td>SITE2 ACCESS2</td>
<td>(,,CN,Beijing,)</td>
<td>(10,PE-Diverse,10)</td>
<td>PE2</td>
<td></td>
</tr>
<tr>
<td>SITE3 ACCESS3</td>
<td>(,,UK,London,)</td>
<td>(12,same-PE,12)</td>
<td>PE4</td>
<td></td>
</tr>
<tr>
<td>SITE4 ACCESS4</td>
<td>(,,FR,Paris,)</td>
<td>(20,Bearer-Diverse,20)</td>
<td>PE7</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Mapping Between VPN Site and VNAP

5. Applicability of TE-Service Mapping in Generic context

As discussed in the Introduction Section, the models presented in this document are also applicable generically outside of the ACTN architecture. [RFC8309] defines Customer Service Model between Customer and Service Orchestrator and Service Delivery Model between Service Orchestrator and Network Orchestrator(s). TE-Service mapping models defined in this document can be regarded primarily as Customer Service Model and secondarily as Service Deliver Model.
6. YANG Data Trees

module: ietf-l1csm-te-service-mapping
augment /l1:l1-connectivity/l1:services/l1:service:
  +rw te-service-mapping!
augment /l1:l1-connectivity/l1:services/l1:service:
  +rw te-mapping
    +rw map-type? identityref
    +rw availability-type? identityref
    +rw (te)?
      +:(actn-vn)
      | +rw actn-vn-ref? -> /vn:vn/vn-list/vn-id
      +:(te-topo)
      | +rw vn-topology-id? te-types:te-topology-id
      | +rw abstract-node? -> /nw:networks/network/node/node-id
      +:(te-tunnel)
      | +rw te-tunnel-list* te:tunnel-ref
augment /l1:l1-connectivity/l1:services/l1:service/l1:endpoint-1:
  +rw (te)?
    +:(actn-vn)
    | +rw actn-vn-ref? -> /vn:ap/access-point-list/access-point-id
    +:(te)
    | +rw ltp? te-types:te-tp-id
augment /l1:l1-connectivity/l1:services/l1:service/l1:endpoint-2:
  +rw (te)?
    +:(actn-vn)
    | +rw actn-vn-ref? -> /vn:ap/access-point-list/access-point-id
    +:(te)
    | +rw ltp? te-types:te-tp-id

module: ietf-l2sm-te-service-mapping
augment /l2vpn-svc:l2vpn-svc/l2vpn-svc:vpn-services/l2vpn-svc:vpn-service:
  +rw te-service-mapping!
augment /l2vpn-svc:l2vpn-svc/l2vpn-svc:vpn-services/l2vpn-svc:vpn-service:
  +rw te-mapping
    +rw map-type? identityref
    +rw availability-type? identityref
    +rw (te)?
      +:(actn-vn)
      | +rw actn-vn-ref? -> /vn:vn/vn-list/vn-id
      +:(te-topo)
      | +rw vn-topology-id? te-types:te-topology-id
      | +rw abstract-node? -> /nw:networks/network/node/node-id
      +:(te-tunnel)
      | +rw te-tunnel-list* te:tunnel-ref
work-accesses/l2vpn-svc:site-network-access:

7. YANG Data Models

The YANG codes are as follows:

```yaml
<CODE BEGINS> file "ietf-te-service-mapping-types@2019-03-05.yang"
module ietf-te-service-mapping-types {
  prefix "tsm";
  import ietf-te-types {
    prefix "te-types";
  }
  import ietf-network {
    prefix "nw";
  }
}

import ietf-te {
    prefix "te";
}

import ietf-vn {
    prefix "vn";
}

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

contact
    "Editor: Young Lee <leeyoung@huawei.com>
    Dhruv Dhody <dhruv.ietf@gmail.com>
    Qin Wu <bill.wu@huawei.com>";

description
    "This module contains a YANG module for TE & Service mapping
parameters and policies as a common grouping applicable to
various service models (e.g., L1CSM, L2SM, L3SM, etc.)";

revision 2019-03-05 {
    description
        "initial version.";
    reference
        "TBD";
}

/ *
* Identity for map-type
*/

identity map-type {
    description
        "Base identity from which specific map types are
derived.";
}

identity new {
    base map-type;
    description
        "The new VN/tunnels are binded to the service.";
}

identity detnet-hard-isolation {
    base new;
    description
        "Hard isolation with deterministic characteristics.";
}
identity hard-isolation {
  base new;
  description
    "Hard isolation."
}

identity soft-isolation {
  base new;
  description
    "Soft-isolation."
}

identity select {
  base map-type;
  description
    "The VPN service selects an existing tunnel with no
     modification."
}

identity modify {
  base map-type;
  description
    "The VPN service selects an existing tunnel and allows
     to modify the properties of the tunnel (e.g., b/w)"
}

/*
 * Identity for availability-type
 */
identity availability-type {
  description
    "Base identity from which specific map types are
     derived."
}

identity level-1 {
  base availability-type;
  description
    "level 1: 99.9999%"
}

identity level-2 {
  base availability-type;
  description
    "level 2: 99.999%"
}
identity level-3 {
    base availability-type;
    description
        "level 3: 99.99%";
}

identity level-4 {
    base availability-type;
    description
        "level 4: 99.9%";
}

identity level-5 {
    base availability-type;
    description
        "level 5: 99%";
}

/*
 * Groupings
 */

grouping te-ref {
    description
        "The reference to TE.";
    choice te {
        description
            "The TE";
        case actn-vn {
            leaf actn-vn-ref {
                type leafref {
                    path "/vn:vn/vn:vn-list/vn:vn-id";
                }
                description
                    "The reference to ACTN VN";
            }
        }
        case te-topo {
            leaf vn-topology-id{
                type te-types:te-topology-id;
                description
                    "An identifier to the TE Topology Model
                     where the abstract nodes and links of
                     the Topology can be found for Type 2
                     VNS";
            }
            leaf abstract-node {
                type leafref {
                    path "/nw:networks/nw:network/nw:node/"
case te-tunnel {
  leaf-list te-tunnel-list {
    type te:tunnel-ref;
    description
    "Reference to TE Tunnels";
  }
}

{p
grouping te-endpoint-ref {
  description
  "The reference to TE endpoints.";
  choice te {
    description
    "The TE";
    case actn-vn {
      leaf actn-vn-ref {
        type leafref {
          path "/vn:ap/vn:access-point-list"
            + "/vn:access-point-id";
        }
        description
        "The reference to ACTN VN";
      }
    }
    case te {
      leaf ltp {
        type te-types:te-tp-id;
        description
        "Reference LTP in the TE-topology";
      }
    }
  }
}

{p
grouping te-mapping {
  description
  "a reference to the abstract node in TE Topology";
  }
}
container te-mapping {
    description "Mapping between Services and TE";
    leaf map-type {
        type identityref {
            base map-type;
        }
        description "Isolation Requirements, Tunnel Bind or Tunnel Selection";
    }
    leaf availability-type {
        type identityref {
            base availability-type;
        }
        description "Availability Requirement for the Service";
    }
    uses te-ref;
}
This module contains a YANG module for the mapping of Layer 1 Connectivity Service Module (L1CSM) to the TE and VN.

Initial version.

Configuration data nodes

This augment is only valid for TE mapping -- te mapping is added.

This augment is only valid for TE mapping -- endpoint-1 te-reference is added.

This augment is only valid for TE mapping -- endpoint-2 te-reference is added.
module ietf-l2sm-te-service-mapping {
  prefix "tm";
  import ietf-te-service-mapping-types {
    prefix "tsm-types";
  }
  import ietf-l2vpn-svc {
    prefix "l2vpn-svc";
  }
  organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS)
     Working Group";
  contact
    "Editor: Young Lee <leeyoung@huawei.com>
      Dhruv Dhody <dhruv.ietf@gmail.com>
      Qin Wu <bill.wu@huawei.com>");
  description
    "This module contains a YANG module for the mapping of
     Layer 2 Service Model (L1CSM) to the TE and VN ";
  revision 2019-03-05 {
    description
      "initial version.";
    reference
      "TBD";
  }
  /*
   * Configuration data nodes
   */
  augment "/l2vpn-svc:l2vpn-svc/l2vpn-svc:vpn-services/l2vpn-svc:vpn-service" {
    description
"l2sm augmented to include TE parameters and mapping";
container te-service-mapping {
  presence "indicates l2 service to te mapping";
  description
    "Container to augment l2sm to TE parameters and mapping";
}

augment "/l2vpn-svc:l2vpn-svc/l2vpn-svc:vpn-services/l2vpn-svc:vpn-service" {
  description
    "This augment is only valid for TE mapping --
    te mapping is added";
  uses tsm-types:te-mapping;
}

  description
    "This augment is only valid for TE mapping --
    network-access te-reference is added";
  uses tsm-types:te-endpoint-ref;
}

<CODE ENDS>

<CODE BEGINS> file "ietf-l3sm-te-service-mapping@2019-03-05.yang"

module ietf-l3sm-te-service-mapping {
  prefix "tm";

  import ietf-te-service-mapping-types {
    prefix "tsm-types";
  }

  import ietf-l3vpn-svc {
    prefix "l3vpn-svc";
  }
}

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

contact
"Editor: Young Lee <leeyoung@huawei.com>
Dhruv Dhody <dhruv.ietf@gmail.com>
Qin Wu <bill.wu@huawei.com>";

description
"This module contains a YANG module for the mapping of
Layer 3 Service Model (L3SM) to the TE and VN ";

revision 2019-03-05 {
  description
   "initial version.";
  reference
   "TBD";
}

/*
 * Configuration data nodes
 */
augment "/l3vpn-svc:l3vpn-svc/l3vpn-svc:vpn-services/l3vpn-svc:vpn-service" {
  description
   "l3sm augmented to include TE parameters and mapping";
  container te-service-mapping {
    presence "indicates l3 service to te mapping";
    description
    "Container to augment l3sm to TE parameters and mapping";
  }
}
augment "/l3vpn-svc:l3vpn-svc/l3vpn-svc:vpn-services/l3vpn-svc:vpn-service" {
  description
   "This augment is only valid for TE mapping --
te mapping is added";
  uses tsm-types:te-mapping;
}
  description
  "This augment is only valid for TE mapping --
  network-access te-reference is added";
  uses tsm-types:te-endpoint-ref;
}
8. Security

The configuration, state, and action data defined in this document are designed to be accessed via a management protocol with a secure transport layer, such as NETCONF [RFC6241]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a preconfigured subset of all available NETCONF protocol operations and content.

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

9. IANA Considerations

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

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

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

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

This document registers the following YANG modules in the YANG Module.

Names registry [RFC7950]:

- name: ietf-te-service-mapping-types
  reference: RFC XXXX (TDB)

- name: ietf-l1csm-te-service-mapping
  reference: RFC XXXX (TDB)

- name: ietf-l2sm-te-service-mapping
  reference: RFC XXXX (TDB)

- name: ietf-l3sm-te-service-mapping
  reference: RFC XXXX (TDB)
10. Acknowledgements

We thank Diego Caviglia and Igor Bryskin for useful discussions and motivation for this work.

11. References

11.1. Informative References


12. Contributors

Adrian Farrel
Old Dog Consulting
Email: adrian@olddog.co.uk

Italo Busi
Huawei Technologies
Email: Italo.Busi@huawei.com

Authors' Addresses

Young Lee
Huawei Technologies
5340 Legacy Drive
Plano, TX 75023, USA
Phone: (469)277-5838
Email: leeyoung@huawei.com

Dhruv Dhody
Huawei Technologies
Email: dhruv.ietf@gmail.com
YANG Data Model for Layer 3 TE Topologies
draft-ietf-teas-yang-l3-te-topo-04

Abstract

This document defines a YANG data model for layer 3 traffic engineering topologies.

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 September 12, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of
1. Introduction

This document defines a YANG [RFC7950] data model for describing the relationship between a layer 3 network topology [RFC8346] and a TE topology [I-D.ietf-teas-yang-te-topo].
When traffic engineering is enabled on a layer 3 network topology, there will be a corresponding TE topology. The TE topology may or may not be congruent to the layer 3 network topology. When such a congruent TE topology exists, there will be a one-to-one association between the one modeling element in the layer 3 topology to another element in the TE topology. When such a congruent TE topology does not exist, the association will not be one-to-one. This YANG data model allows both cases.

1.1. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119].

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

- augment
- data model
- data node

1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Modeling Considerations for L3 TE Topologies

2.1. Relationship Between Layer 3 Topology and TE topology

In general, layer 3 network topology model and TE topology model can be used independently. When traffic engineering is enabled on a layer 3 network topology, there will be associations between objects in layer 3 network topologies and objects in TE topologies. The properties of these relations are:

- The associations are between objects of the same class, i.e. node to node or link to link.

- The multiplicity of such an association is: 0..1 to 0..1. An object in a layer 3 network may have zero or one associated object in the corresponding TE network.
2.2. Relationship Modeling

YANG data type leafref is used to model the association relationship between a layer 3 network topology and a TE topology. YANG must statements are used to enforce the referenced objects are in the topologies of proper type.

2.2.1. Topology Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, the layer 3 network topology will have a reference to the corresponding TE topology. Such a reference is modeled as follows:

```yang
augment /nw:networks/nw:network/l3t:l3-topology-attributes:
  +--rw l3-te-topology-attributes
```

If the TE topology is congruent to the layer 3 network topology, the above reference can still be used to specified TE paramenters defined in the TE topology model.

2.2.2. Node Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 network node may have a reference to the corresponding TE node. Such a reference is modeled as follows:

```yang
augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
  +--rw l3-te-node-attributes
    +--rw node-ref?      leafref
```

2.2.3. Link Termination Point Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 link termination point may have a reference to the corresponding TE link termination point. Such a reference is modeled as follows:

```yang
augment /nw:networks/nw:network/nw:node/nt:termination-point
  /l3t:l3-termination-point-attributes:
    +--rw l3-te-tp-attributes
      +--rw tp-ref?        leafref
      +--rw node-ref?      leafref
```
2.2.4. Link Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 link may have a reference to the corresponding TE link. Such a reference is modeled as follows:

```
augment /nw:networks/nw:network/nt:link/l3t:l3-link-attributes:
   +--rw l3-te-link-attributes
      +--rw link-ref?    leafref
```

2.3. Topology Type Modeling

A new topology type is defined in this document, to indicate a topology that is a layer 3 topology with TE enabled.

```
augment /nw:networks/nw:network/nw:network-types
   /l3t:l3-unicast-topology:
      +--rw l3-te!
```

3. Packet Switching Technology Extensions

3.1. Technology Specific Link Attributes

The technology agnostic TE Topology model is augmented with packet switching specific link attributes:

```
   /tet:te-link-attributes 
   /tet:interface-switching-capability:
      +--rw packet-switch-capable
      +--rw minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
      +--rw interface-mtu?         uint16
```

```
augment /nw:networks/nw:network/nt:link/tet:te 
   /tet:te-link-attributes 
   /tet:interface-switching-capability:
      +--rw packet-switch-capable
      +--rw minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
      +--rw interface-mtu?         uint16
```

```
augment /nw:networks/nw:network/nt:link/tet:te 
   /tet:information-source-entry 
   /tet:interface-switching-capability:
      +--ro packet-switch-capable
      +--ro minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
      +--ro interface-mtu?         uint16
```
3.2. Performance Metric

[RFC7471], [RFC7810] and [RFC7823] specify TE performance metric parameters and their usage. The packet switching augmentations specified in this document support such a capability, which can be conditional enabled by a YANG feature "te-performance-metric".

```
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices:
    +--rw performance-metric
        +--rw measurement
        |     ......
        +--rw normality
        |     ......
        +--rw throttle
     ......
```

Such an augmentation has been applied to:

- Connectivity matrices container
- Connectivity matrix entry
- Local ink connectivities container
- Local ink connectivity entry
- TE link attributes container in a TE link template
- TE link attributes container in a TE link
- Information source entry in a TE link

4. Model Structure

4.1. Layer 3 TE Topology Module

The model tree structure of the layer 3 TE topology module is as shown below:
4.2. Packet Switching TE Topology Module

This is an augmentation to base TE topology model.
te-types:performance-metrics-normality

++-ro one-way-min-delay?            uint32
++-ro one-way-min-delay-normality?
    te-types:performance-metrics-normality
++-ro one-way-max-delay?            uint32
++-ro one-way-max-delay-normality?
    te-types:performance-metrics-normality
++-ro one-way-delay-variation?      uint32
++-ro one-way-delay-variation-normality?
    te-types:performance-metrics-normality
++-ro one-way-packet-loss?          decimal64
++-ro one-way-packet-loss-normality?
    te-types:performance-metrics-normality
++-ro performance-metrics-two-way
++-ro two-way-delay?                uint32
++-ro two-way-delay-normality?
    te-types:performance-metrics-normality
++-ro two-way-min-delay?            uint32
++-ro two-way-min-delay-normality?
    te-types:performance-metrics-normality
++-ro two-way-max-delay?            uint32
++-ro two-way-max-delay-normality?
    te-types:performance-metrics-normality
++-ro two-way-delay-variation?      uint32
++-ro two-way-delay-variation-normality?
    te-types:performance-metrics-normality
++-ro two-way-packet-loss?          decimal64
++-ro two-way-packet-loss-normality?
    te-types:performance-metrics-normality

++-rw throttle
++-rw one-way-delay-offset?         uint32
++-rw measure-interval?             uint32
++-rw advertisement-interval?       uint32
++-rw suppression-interval?         uint32
++-rw threshold-out
    ++-rw one-way-delay?                uint32
    ++-rw one-way-residual-bandwidth?
        rt-types:bandwidth-ieee-float32
    ++-rw one-way-available-bandwidth?
        rt-types:bandwidth-ieee-float32
    ++-rw one-way-utilized-bandwidth?
        rt-types:bandwidth-ieee-float32
    ++-rw two-way-delay?                uint32
    ++-rw one-way-min-delay?            uint32
    ++-rw one-way-max-delay?            uint32
    ++-rw one-way-delay-variation?      uint32
    ++-rw one-way-packet-loss?          decimal64
    ++-rw two-way-min-delay?            uint32

Internet-Draft             YANG L3 TE Topology                March 2019

|  +--rw two-way-max-delay?             uint32
|  +--rw two-way-delay-variation?       uint32
|  +--rw two-way-packet-loss?           decimal64
++--rw threshold-in
    |  +--rw one-way-delay?                 uint32
    |  +--rw one-way-residual-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw one-way-available-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw one-way-utilized-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw two-way-delay?                 uint32
    |  +--rw one-way-min-delay?             uint32
    |  +--rw one-way-max-delay?             uint32
    |  +--rw one-way-delay-variation?       uint32
    |  +--rw one-way-packet-loss?           decimal64
    |  +--rw two-way-min-delay?             uint32
    |  +--rw two-way-max-delay?             uint32
    |  +--rw two-way-delay-variation?       uint32
    |  +--rw two-way-packet-loss?           decimal64
+++--rw threshold-accelerated-advertisement
    |  +--rw one-way-delay?                 uint32
    |  +--rw one-way-residual-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw one-way-available-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw one-way-utilized-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--rw two-way-delay?                 uint32
    |  +--rw one-way-min-delay?             uint32
    |  +--rw one-way-max-delay?             uint32
    |  +--rw one-way-delay-variation?       uint32
    |  +--rw one-way-packet-loss?           decimal64
    |  +--rw two-way-min-delay?             uint32
    |  +--rw two-way-max-delay?             uint32
    |  +--rw two-way-delay-variation?       uint32
    |  +--rw two-way-packet-loss?           decimal64

augment /nw:networks/nw:network/nw:node/tet:te
     /tet:te-node-attributes/tet:connectivity-matrices
     /tet:connectivity-matrix:
+++--ro performance-metrics-one-way
    |  +--ro one-way-delay?                 uint32
    |  +--ro one-way-delay-normality?
    |     te-types:performance-metrics-normality
    |  +--ro one-way-residual-bandwidth?
    |     rt-types:bandwidth-ieee-float32
    |  +--ro one-way-residual-bandwidth-normality?
        te-types:performance-metrics-normality

---ro one-way-available-bandwidth?
  rttypes:bandwidth-ieee-float32
---ro one-way-available-bandwidth-normality?
  te-types:performance-metrics-normality
---ro one-way-utilized-bandwidth?
  rttypes:bandwidth-ieee-float32
---ro one-way-utilized-bandwidth-normality?
  te-types:performance-metrics-normality
---ro one-way-min-delay? uint32
---ro one-way-min-delay-normality?
  te-types:performance-metrics-normality
---ro one-way-max-delay? uint32
---ro one-way-max-delay-normality?
  te-types:performance-metrics-normality
---ro one-way-delay-variation? uint32
---ro one-way-delay-variation-normality?
  te-types:performance-metrics-normality
---ro one-way-packet-loss? decimal64
---ro one-way-packet-loss-normality?
  te-types:performance-metrics-normality
---ro performance-metrics-two-way
  ---ro two-way-delay? uint32
  ---ro two-way-delay-normality?
  te-types:performance-metrics-normality
  ---ro two-way-min-delay? uint32
  ---ro two-way-min-delay-normality?
  te-types:performance-metrics-normality
  ---ro two-way-max-delay? uint32
  ---ro two-way-max-delay-normality?
  te-types:performance-metrics-normality
  ---ro two-way-delay-variation? uint32
  ---ro two-way-delay-variation-normality?
  te-types:performance-metrics-normality
  ---ro two-way-packet-loss? decimal64
  ---ro two-way-packet-loss-normality?
  te-types:performance-metrics-normality
---rw throttle
  ---rw one-way-delay-offset? uint32
  ---rw measure-interval? uint32
  ---rw advertisement-interval? uint32
  ---rw suppression-interval? uint32
  ---rw threshold-out
    ---rw one-way-delay? uint32
    ---rw one-way-residual-bandwidth?
      rttypes:bandwidth-ieee-float32
    ---rw one-way-available-bandwidth?
      rttypes:bandwidth-ieee-float32
    ---rw one-way-utilized-bandwidth?
Internet-Draft             YANG L3 TE Topology                March 2019

| rt-types:bandwidth-ieee-float32
| +--rw two-way-delay?     uint32
| +--rw one-way-min-delay? uint32
| +--rw one-way-max-delay? uint32
| +--rw one-way-delay-variation? uint32
| +--rw one-way-packet-loss? decimal64
| +--rw two-way-min-delay? uint32
| +--rw two-way-max-delay? uint32
| +--rw two-way-delay-variation? uint32
| +--rw two-way-packet-loss? decimal64
++--rw threshold-in
| +--rw one-way-delay?     uint32
| +--rw one-way-residual-bandwidth? rt-types:bandwidth-ieee-float32
| | +--rw one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
| | +--rw one-way-utilized-bandwidth? rt-types:bandwidth-ieee-float32
| +--rw two-way-delay?     uint32
| +--rw one-way-min-delay? uint32
| +--rw one-way-max-delay? uint32
| +--rw one-way-delay-variation? uint32
| +--rw one-way-packet-loss? decimal64
| +--rw two-way-min-delay? uint32
| +--rw two-way-max-delay? uint32
| +--rw two-way-delay-variation? uint32
| +--rw two-way-packet-loss? decimal64
++--rw threshold-accelerated-advertisement
| +--rw one-way-delay?     uint32
| +--rw one-way-residual-bandwidth? rt-types:bandwidth-ieee-float32
| | +--rw one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
| | +--rw one-way-utilized-bandwidth? rt-types:bandwidth-ieee-float32
| +--rw two-way-delay?     uint32
| +--rw one-way-min-delay? uint32
| +--rw one-way-max-delay? uint32
| +--rw one-way-delay-variation? uint32
| +--rw one-way-packet-loss? decimal64
| +--rw two-way-min-delay? uint32
| +--rw two-way-max-delay? uint32
| +--rw two-way-delay-variation? uint32
| +--rw two-way-packet-loss? decimal64
| +--ro performance-metrics-one-way
| | +--ro one-way-delay?     uint32
++ ro one-way-delay-normality?
  te-types:performance-metrics-normality
++ ro one-way-residual-bandwidth?
  rt-types:bandwidth-ieee-float32
++ ro one-way-residual-bandwidth-normality?
  te-types:performance-metrics-normality
++ ro one-way-available-bandwidth?
  rt-types:bandwidth-ieee-float32
++ ro one-way-available-bandwidth-normality?
  te-types:performance-metrics-normality
++ ro one-way-utilized-bandwidth?
  rt-types:bandwidth-ieee-float32
++ ro one-way-utilized-bandwidth-normality?
  te-types:performance-metrics-normality
++ ro one-way-min-delay?     uint32
  te-types:performance-metrics-normality
++ ro one-way-max-delay?     uint32
  te-types:performance-metrics-normality
++ ro one-way-delay-variation?     uint32
  te-types:performance-metrics-normality
++ ro one-way-packet-loss?     decimal64
  te-types:performance-metrics-normality
++ ro performance-metrics-two-way
  ++ ro two-way-delay?     uint32
  te-types:performance-metrics-normality
++ ro two-way-delay-normality?
  te-types:performance-metrics-normality
++ ro two-way-min-delay?     uint32
  te-types:performance-metrics-normality
++ ro two-way-max-delay?     uint32
  te-types:performance-metrics-normality
++ ro two-way-max-delay-normality?
  te-types:performance-metrics-normality
++ ro two-way-delay-variation?     uint32
  te-types:performance-metrics-normality
++ ro two-way-delay-variation-normality?
  te-types:performance-metrics-normality
++ ro two-way-packet-loss?     decimal64
  te-types:performance-metrics-normality
++ ro two-way-packet-loss-normality?
  te-types:performance-metrics-normality
++ ro throttle
  ++ ro one-way-delay-offset?     uint32
++ ro measure-interval?     uint32
++ ro advertisement-interval?     uint32
++ ro suppression-interval?     uint32
++ ro threshold-out
Internet-Draft             YANG L3 TE Topology                March 2019
|  +--ro one-way-delay?                 uint32
|  +--ro one-way-residual-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-available-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-utilized-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro two-way-delay?                 uint32
|  +--ro one-way-min-delay?             uint32
|  +--ro one-way-max-delay?             uint32
|  +--ro one-way-delay-variation?       uint32
|  +--ro one-way-packet-loss?           decimal64
|  +--ro two-way-min-delay?             uint32
|  +--ro two-way-max-delay?             uint32
|  +--ro two-way-delay-variation?       uint32
|  +--ro two-way-packet-loss?           decimal64
++--ro threshold-in
|  +--ro one-way-delay?                 uint32
|  +--ro one-way-residual-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-available-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-utilized-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro two-way-delay?                 uint32
|  +--ro one-way-min-delay?             uint32
|  +--ro one-way-max-delay?             uint32
|  +--ro one-way-delay-variation?       uint32
|  +--ro one-way-packet-loss?           decimal64
|  +--ro two-way-min-delay?             uint32
|  +--ro two-way-max-delay?             uint32
|  +--ro two-way-delay-variation?       uint32
|  +--ro two-way-packet-loss?           decimal64
++--ro threshold-accelerated-advertisement
|  +--ro one-way-delay?                 uint32
|  +--ro one-way-residual-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-available-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro one-way-utilized-bandwidth?
|      rt-types:bandwidth-ieee-float32
|  +--ro two-way-delay?                 uint32
|  +--ro one-way-min-delay?             uint32
|  +--ro one-way-max-delay?             uint32
|  +--ro one-way-delay-variation?       uint32
|  +--ro one-way-packet-loss?           decimal64
|  +--ro two-way-min-delay?             uint32
|  +--ro two-way-max-delay?             uint32
|  +--ro two-way-delay-variation?       uint32
|  +--ro two-way-packet-loss?           decimal64

  ++--ro performance-metrics-one-way
    ++--ro one-way-delay? uint32
    ++--ro one-way-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-delay-normality?              te-types:performance-metrics-normality
    ++--ro one-way-residual-bandwidth?
      |     rt-types:bandwidth-ieee-float32
    ++--ro one-way-residual-bandwidth-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-available-bandwidth?
      |     rt-types:bandwidth-ieee-float32
    ++--ro one-way-available-bandwidth-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-utilized-bandwidth?
      |     rt-types:bandwidth-ieee-float32
    ++--ro one-way-utilized-bandwidth-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-min-delay?                uint32
    ++--ro one-way-min-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-max-delay?                uint32
    ++--ro one-way-max-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-delay-variation?          uint32
    ++--ro one-way-delay-variation-normality?
      |     te-types:performance-metrics-normality
    ++--ro one-way-packet-loss?            decimal64
    ++--ro one-way-packet-loss-normality?
      |     te-types:performance-metrics-normality
  ++--ro performance-metrics-two-way
    ++--ro two-way-delay?        uint32
    ++--ro two-way-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro two-way-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro two-way-min-delay?               uint32
    ++--ro two-way-min-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro two-way-max-delay?               uint32
    ++--ro two-way-max-delay-normality?
      |     te-types:performance-metrics-normality
    ++--ro two-way-delay-variation?       uint32
    ++--ro two-way-delay-variation-normality?
      |     te-types:performance-metrics-normality
    ++--ro two-way-packet-loss?            decimal64
    ++--ro two-way-packet-loss-normality?
    ++--ro two-way-packet-loss-normality?
te-types:performance-metrics-normality

++-ro throttle
  +++-ro one-way-delay-offset? uint32
  +++-ro measure-interval? uint32
  +++-ro advertisement-interval? uint32
  +++-ro suppression-interval? uint32

++-ro threshold-out
  +++-ro one-way-delay? uint32
    |  +++-ro one-way-residual-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-available-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-utilized-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro two-way-delay? uint32
    |  +++-ro one-way-min-delay? uint32
    |  +++-ro one-way-max-delay? uint32
    |  +++-ro one-way-delay-variation? uint32
    |  +++-ro one-way-packet-loss? decimal64
    |  +++-ro two-way-min-delay? uint32
    |  +++-ro two-way-max-delay? uint32
    |  +++-ro two-way-delay-variation? uint32
    |  +++-ro two-way-packet-loss? decimal64

++-ro threshold-in
  +++-ro one-way-delay? uint32
    |  +++-ro one-way-residual-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-available-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-utilized-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro two-way-delay? uint32
    |  +++-ro one-way-min-delay? uint32
    |  +++-ro one-way-max-delay? uint32
    |  +++-ro one-way-delay-variation? uint32
    |  +++-ro one-way-packet-loss? decimal64
    |  +++-ro two-way-min-delay? uint32
    |  +++-ro two-way-max-delay? uint32
    |  +++-ro two-way-delay-variation? uint32
    |  +++-ro two-way-packet-loss? decimal64

++-ro threshold-accelerated-advertisement
  +++-ro one-way-delay? uint32
    |  +++-ro one-way-residual-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-available-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
    |  +++-ro one-way-utilized-bandwidth?
    |    |  rt-types:bandwidth-ieee-float32
+--ro two-way-delay?     uint32
+--ro one-way-min-delay? uint32
+--ro one-way-max-delay? uint32
+--ro one-way-delay-variation? uint32
+--ro one-way-packet-loss? decimal64
+--ro two-way-min-delay? uint32
+--ro two-way-max-delay? uint32
+--ro two-way-delay-variation? uint32
+--ro two-way-packet-loss? decimal64

augment /nw:networks/nw:network/nw:node/tet:te
/tet:tunnel-termination-point
/tet:local-link-connectivities:

+--ro performance-metrics-one-way
  +--ro one-way-delay?     uint32
  +--ro one-way-delay-normality?  
      te-types:performance-metrics-normality
  +--ro one-way-residual-bandwidth? rt-types:bandwidth-ieee-float32
  +--ro one-way-residual-bandwidth-normality? 
      te-types:performance-metrics-normality
  +--ro one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
  +--ro one-way-available-bandwidth-normality? 
      te-types:performance-metrics-normality
  +--ro one-way-utilized-bandwidth? rt-types:bandwidth-ieee-float32
  +--ro one-way-utilized-bandwidth-normality? 
      te-types:performance-metrics-normality
  +--ro one-way-min-delay?     uint32
  +--ro one-way-min-delay-normality?  
      te-types:performance-metrics-normality
  +--ro one-way-max-delay?     uint32
  +--ro one-way-max-delay-normality?  
      te-types:performance-metrics-normality
  +--ro one-way-delay-variation? uint32
  +--ro one-way-delay-variation-normality? 
      te-types:performance-metrics-normality
  +--ro one-way-packet-loss?    decimal64
  +--ro one-way-packet-loss-normality? 
      te-types:performance-metrics-normality

+--ro performance-metrics-two-way
  +--ro two-way-delay?     uint32
  +--ro two-way-delay-normality?  
      te-types:performance-metrics-normality
  +--ro two-way-min-delay?     uint32
  +--ro two-way-min-delay-normality?  
      te-types:performance-metrics-normality
  +--ro two-way-max-delay?     uint32
  +--ro two-way-max-delay-normality?  
      te-types:performance-metrics-normality
++-ro two-way-max-delay-normality?
   te-types:performance-metrics-normality
++-ro two-way-delay-variation?
   uint32
++-ro two-way-delay-variation-normality?
   te-types:performance-metrics-normality
++-ro two-way-packet-loss?
   decimal64
++-ro two-way-packet-loss-normality?
   te-types:performance-metrics-normality
++-rw throttle
   +++-rw one-way-delay-offset?    uint32
   +++-rw measure-interval?      uint32
   +++-rw advertisement-interval? uint32
   +++-rw suppression-interval?  uint32
   +++-rw threshold-out
      +++-rw one-way-delay?    uint32
      +++-rw one-way-residual-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw one-way-available-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw one-way-utilized-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw two-way-delay?    uint32
      +++-rw one-way-min-delay?  uint32
      +++-rw one-way-max-delay?  uint32
      +++-rw one-way-delay-variation?  uint32
      +++-rw one-way-packet-loss?  decimal64
      +++-rw two-way-min-delay?  uint32
      +++-rw two-way-max-delay?  uint32
      +++-rw two-way-delay-variation?  uint32
      +++-rw two-way-packet-loss?  decimal64
    +++-rw threshold-in
      +++-rw one-way-delay?    uint32
      +++-rw one-way-residual-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw one-way-available-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw one-way-utilized-bandwidth?
         rt-types:bandwidth-ieee-float32
      +++-rw two-way-delay?    uint32
      +++-rw one-way-min-delay?  uint32
      +++-rw one-way-max-delay?  uint32
      +++-rw one-way-delay-variation?  uint32
      +++-rw one-way-packet-loss?  decimal64
      +++-rw two-way-min-delay?  uint32
      +++-rw two-way-max-delay?  uint32
      +++-rw two-way-delay-variation?  uint32
      +++-rw two-way-packet-loss?  decimal64
   +++-rw threshold-accelerated-advertisement
++-rw one-way-delay?      uint32
++-rw one-way-residual-bandwidth?
  |      rt-types:bandwidth-ieee-float32
++-rw one-way-available-bandwidth?
  |      rt-types:bandwidth-ieee-float32
++-rw one-way-utilized-bandwidth?
  |      rt-types:bandwidth-ieee-float32
++-rw two-way-delay?      uint32
++-rw one-way-min-delay?   uint32
++-rw one-way-max-delay?   uint32
++-rw one-way-delay-variation?  uint32
++-rw one-way-packet-loss? decimal64
++-rw two-way-min-delay?   uint32
++-rw two-way-max-delay?   uint32
++-rw two-way-delay-variation?  uint32
++-rw two-way-packet-loss? decimal64

augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity:

  +++-ro performance-metrics-one-way
  |    +++-ro one-way-delay?      uint32
  |    +++-ro one-way-delay-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-residual-bandwidth?
  |      rt-types:bandwidth-ieee-float32
  |    +++-ro one-way-residual-bandwidth-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-available-bandwidth?
  |      rt-types:bandwidth-ieee-float32
  |    +++-ro one-way-available-bandwidth-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-utilized-bandwidth?
  |      rt-types:bandwidth-ieee-float32
  |    +++-ro one-way-utilized-bandwidth-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-min-delay?   uint32
  |    +++-ro one-way-min-delay-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-max-delay?   uint32
  |    +++-ro one-way-max-delay-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-delay-variation?  uint32
  |    +++-ro one-way-delay-variation-normality?
  |      te-types:performance-metrics-normality
  |    +++-ro one-way-packet-loss? decimal64
  |    +++-ro one-way-packet-loss-normality?
  |      te-types:performance-metrics-normality
++--ro performance-metrics-two-way
  +-++--ro two-way-delay?                  uint32
  +-++--ro two-way-delay-normality?
    |      te-types:performance-metrics-normality
  +-++--ro two-way-min-delay?              uint32
  +-++--ro two-way-min-delay-normality?
    |      te-types:performance-metrics-normality
  +-++--ro two-way-max-delay?              uint32
  +-++--ro two-way-max-delay-normality?
    |      te-types:performance-metrics-normality
  +-++--ro two-way-delay-variation?        uint32
  +-++--ro two-way-delay-variation-normality?
    |      te-types:performance-metrics-normality
  +-++--ro two-way-packet-loss?           decimal64
    |      te-types:performance-metrics-normality
  +-++--rw throttle
    +-++--rw one-way-delay-offset?         uint32
    +-++--rw measure-interval?             uint32
    +-++--rw advertisement-interval?       uint32
    +-++--rw suppression-interval?         uint32
    +-++--rw threshold-out
      +-++--rw one-way-delay?               uint32
      +-++--rw one-way-residual-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw one-way-available-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw one-way-utilized-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw two-way-delay?               uint32
      +-++--rw one-way-min-delay?            uint32
      +-++--rw one-way-max-delay?            uint32
      +-++--rw one-way-delay-variation?      uint32
      +-++--rw one-way-packet-loss?         decimal64
      +-++--rw two-way-min-delay?            uint32
      +-++--rw two-way-max-delay?            uint32
      +-++--rw two-way-delay-variation?      uint32
      +-++--rw two-way-packet-loss?         decimal64
    +-++--rw threshold-in
      +-++--rw one-way-delay?               uint32
      +-++--rw one-way-residual-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw one-way-available-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw one-way-utilized-bandwidth?
        |      rt-types:bandwidth-ieee-float32
      +-++--rw two-way-delay?               uint32
      +-++--rw one-way-min-delay?            uint32
---rw one-way-max-delay? uint32
---rw one-way-delay-variation? uint32
---rw one-way-packet-loss? decimal64
---rw two-way-min-delay? uint32
---rw two-way-max-delay? uint32
---rw two-way-delay-variation? uint32
---rw two-way-packet-loss? decimal64
---rw threshold-accelerated-advertisement
  ---rw one-way-delay? uint32
  ---rw one-way-residual-bandwidth? rt-types:bandwidth-ieee-float32
  ---rw one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
  ---rw one-way-utilized-bandwidth? rt-types:bandwidth-ieee-float32
  ---rw two-way-delay? uint32
  ---rw one-way-min-delay? uint32
  ---rw one-way-max-delay? uint32
  ---rw one-way-delay-variation? uint32
  ---rw one-way-packet-loss? decimal64
  ---rw two-way-min-delay? uint32
  ---rw two-way-max-delay? uint32
  ---rw two-way-delay-variation? uint32
  ---rw two-way-packet-loss? decimal64

/tet:te-link-attributes:
  ---ro performance-metrics-one-way
    ---ro one-way-delay? uint32
    ---ro one-way-delay-normality? te-types:performance-metrics-normality
    ---ro one-way-residual-bandwidth? rt-types:bandwidth-ieee-float32
    ---ro one-way-residual-bandwidth-normality? te-types:performance-metrics-normality
    ---ro one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
    ---ro one-way-available-bandwidth-normality? te-types:performance-metrics-normality
    ---ro one-way-utilized-bandwidth? rt-types:bandwidth-ieee-float32
    ---ro one-way-utilized-bandwidth-normality? te-types:performance-metrics-normality
    ---ro one-way-min-delay? uint32
    ---ro one-way-min-delay-normality? te-types:performance-metrics-normality
    ---ro one-way-max-delay? uint32
    ---ro one-way-max-delay-normality? te-types:performance-metrics-normality

Internet-Draft             YANG L3 TE Topology                March 2019

---ro one-way-delay-variation?          uint32
+-ro one-way-delay-variation-normality?
|        te-types:performance-metrics-normality
+-ro one-way-packet-loss?              decimal64
+-ro one-way-packet-loss-normality?
|        te-types:performance-metrics-normality

---ro performance-metrics-two-way
---ro two-way-delay?                    uint32
+-ro two-way-delay-normality?
|        te-types:performance-metrics-normality
+-ro two-way-min-delay?                 uint32
+-ro two-way-min-delay-normality?
|        te-types:performance-metrics-normality
+-ro two-way-max-delay?                 uint32
+-ro two-way-max-delay-normality?
|        te-types:performance-metrics-normality
+-ro two-way-delay-variation?           uint32
|        te-types:performance-metrics-normality
+-ro two-way-delay-variation-normality?
|        te-types:performance-metrics-normality
+-ro two-way-packet-loss?              decimal64
+-ro two-way-packet-loss-normality?
|        te-types:performance-metrics-normality

---rw throttle
---rw throttle?
|        throttle?
|        throttle?
|        throttle?

---rw throttle-out
---rw one-way-delay-offset?             uint32
---rw measure-interval?                 uint32
---rw advertisement-interval?           uint32
---rw suppress-interval?                uint32
---rw threshold-out
|        one-way-delay?
|        one-way-residual-bandwidth?
|        rt-types:bandwidth-ieee-float32
|        one-way-available-bandwidth?
|        rt-types:bandwidth-ieee-float32
|        one-way-utilized-bandwidth?
|        rt-types:bandwidth-ieee-float32
---rw two-way-delay?                    uint32
---rw one-way-min-delay?                uint32
---rw one-way-max-delay?                uint32
---rw one-way-delay-variation?          uint32
---rw one-way-delay-variation?
|        decimal64
---rw one-way-packet-loss?              decimal64
---rw one-way-packet-loss?
|        decimal64
---rw two-way-delay?
---rw two-way-max-delay?
---rw two-way-delay-variation?
---rw two-way-delay-variation?
---rw two-way-packet-loss?
---rw two-way-packet-loss?

---rw threshold-in
|        one-way-delay?
|        one-way-residual-bandwidth?
|        rt-types:bandwidth-ieee-float32
++rw one-way-available-bandwidth?
  | rt-types:bandwidth-ieee-float32
++rw one-way-utilized-bandwidth?
  | rt-types:bandwidth-ieee-float32
++rw two-way-delay?                              uint32
++rw one-way-min-delay?                          uint32
++rw one-way-max-delay?                          uint32
++rw one-way-delay-variation?                    uint32
++rw one-way-packet-loss?                        decimal64
++rw two-way-min-delay?                          uint32
++rw two-way-max-delay?                          uint32
++rw two-way-delay-variation?                    uint32
++rw two-way-packet-loss?                        decimal64
++rw threshold-accelerated-advertisement
  ++rw one-way-delay?                              uint32
  ++rw one-way-residual-bandwidth?
    | rt-types:bandwidth-ieee-float32
++rw two-way-delay?                              uint32
++rw one-way-residual-bandwidth?
  | rt-types:bandwidth-ieee-float32
++rw one-way-available-bandwidth?
  | rt-types:bandwidth-ieee-float32
++rw one-way-utilized-bandwidth?
  | rt-types:bandwidth-ieee-float32
++rw two-way-delay?                              uint32
++rw one-way-min-delay?                          uint32
++rw one-way-max-delay?                          uint32
++rw one-way-delay-variation?                    uint32
++rw one-way-packet-loss?                        decimal64
++rw two-way-min-delay?                          uint32
++rw two-way-max-delay?                          uint32
++rw two-way-delay-variation?                    uint32
++rw two-way-packet-loss?                        decimal64
  augment /nw:networks/nw:network/nt:link/tet:te
   /tet:te-link-attributes:
   ++ro performance-metrics-one-way
     ++ro one-way-delay?                              uint32
     ++ro one-way-delay-normality?
       | te-types:performance-metrics-normality
     ++ro one-way-residual-bandwidth?
       | rt-types:bandwidth-ieee-float32
     ++ro one-way-residual-bandwidth-normality?
       | te-types:performance-metrics-normality
     ++ro one-way-available-bandwidth?
       | rt-types:bandwidth-ieee-float32
     ++ro one-way-available-bandwidth-normality?
       | te-types:performance-metrics-normality
     ++ro one-way-utilized-bandwidth?
       | rt-types:bandwidth-ieee-float32
     ++ro one-way-utilized-bandwidth-normality?
       | te-types:performance-metrics-normality
```yang
+--ro one-way-min-delay?  uint32
+--ro one-way-min-delay-normality?
    te-types:performance-metrics-normality
+--ro one-way-max-delay?  uint32
+--ro one-way-max-delay-normality?
    te-types:performance-metrics-normality
+--ro one-way-delay-variation?  uint32
+--ro one-way-delay-variation-normality?
    te-types:performance-metrics-normality
+--ro one-way-packet-loss?  decimal64
+--ro one-way-packet-loss-normality?
    te-types:performance-metrics-normality

+--ro performance-metrics-two-way
+--ro two-way-delay?  uint32
+--ro two-way-delay-normality?
    te-types:performance-metrics-normality
+--ro two-way-min-delay?  uint32
+--ro two-way-min-delay-normality?
    te-types:performance-metrics-normality
+--ro two-way-max-delay?  uint32
+--ro two-way-max-delay-normality?
    te-types:performance-metrics-normality
+--ro two-way-delay-variation?  uint32
+--ro two-way-delay-variation-normality?
    te-types:performance-metrics-normality
+--ro two-way-packet-loss?  decimal64
+--ro two-way-packet-loss-normality?
    te-types:performance-metrics-normality

+--rw throttle
+--rw one-way-delay-offset?  uint32
+--rw measure-interval?  uint32
+--rw advertisement-interval?  uint32
+--rw suppression-interval?  uint32

+--rw threshold-out
+--rw one-way-delay?  uint32
+--rw one-way-residual-bandwidth?
    rt-types:bandwidth-ieee-float32
+--rw one-way-available-bandwidth?
    rt-types:bandwidth-ieee-float32
+--rw one-way-utilized-bandwidth?
    rt-types:bandwidth-ieee-float32
+--rw two-way-delay?  uint32
+--rw one-way-min-delay?  uint32
+--rw one-way-max-delay?  uint32
+--rw one-way-delay-variation?  uint32
+--rw one-way-packet-loss?  decimal64
+--rw two-way-min-delay?  uint32
+--rw two-way-max-delay?  uint32
```

++rw two-way-delay-variation?          uint32
++rw two-way-packet-loss?             decimal64
++rw threshold-in
  +--rw one-way-delay?                 uint32
  +--rw one-way-residual-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw one-way-available-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw one-way-utilized-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw two-way-delay?                 uint32
  +--rw one-way-min-delay?             uint32
  +--rw one-way-max-delay?             uint32
  +--rw one-way-delay-variation?       uint32
  +--rw one-way-packet-loss?           decimal64
  +--rw two-way-min-delay?             uint32
  +--rw two-way-max-delay?             uint32
  +--rw two-way-delay-variation?       uint32
  +--rw two-way-packet-loss?           decimal64
++rw threshold-accelerated-advertisement
  +--rw one-way-delay?                 uint32
  +--rw one-way-residual-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw one-way-available-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw one-way-utilized-bandwidth?
    |   rt-types:bandwidth-ieee-float32
  +--rw two-way-delay?                 uint32
  +--rw one-way-min-delay?             uint32
  +--rw one-way-max-delay?             uint32
  +--rw one-way-delay-variation?       uint32
  +--rw one-way-packet-loss?           decimal64
  +--rw two-way-min-delay?             uint32
  +--rw two-way-max-delay?             uint32
  +--rw two-way-delay-variation?       uint32
  +--rw two-way-packet-loss?           decimal64
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry:
    +--ro performance-metrics-one-way
      +--ro one-way-delay?                 uint32
      +--ro one-way-delay-normality?
        |   te-types:performance-metrics-normality
      +--ro one-way-residual-bandwidth?
        |   rt-types:bandwidth-ieee-float32
      +--ro one-way-residual-bandwidth-normality?
        |   te-types:performance-metrics-normality
      +--ro one-way-available-bandwidth?
        |   rt-types:bandwidth-ieee-float32
---ro one-way-available-bandwidth-normality?
  |   te-types:performance-metrics-normality
---ro one-way-utilized-bandwidth?
  |   rt-types:bandwidth-ieee-float32
---ro one-way-utilized-bandwidth-normality?
  |   te-types:performance-metrics-normality
---ro one-way-min-delay?                       uint32
---ro one-way-min-delay-normality?
  |       te-types:performance-metrics-normality
---ro one-way-max-delay?                       uint32
---ro one-way-max-delay-normality?
  |       te-types:performance-metrics-normality
---ro one-way-delay-variation?                 uint32
---ro one-way-delay-variation-normality?
  |       te-types:performance-metrics-normality
---ro one-way-packet-loss?                     decimal64
---ro one-way-packet-loss-normality?
  |       te-types:performance-metrics-normality
---ro performance-metrics-two-way
  |   uint32
---ro two-way-delay?
---ro two-way-delay-normality?
  |       te-types:performance-metrics-normality
---ro two-way-min-delay?
  |       te-types:performance-metrics-normality
---ro two-way-max-delay?
  |       te-types:performance-metrics-normality
---ro two-way-delay-variation?
  |       te-types:performance-metrics-normality
---ro two-way-packet-loss?                     decimal64
---ro two-way-packet-loss-normality?
  |       te-types:performance-metrics-normality
---ro throttle
  |       uint32
---ro one-way-delay-offset?
---ro measure-interval?
---ro advertisement-interval?
---ro suppression-interval?
---ro threshold-out
  |   uint32
---ro one-way-delay?
---ro one-way-residual-bandwidth?
  |       rt-types:bandwidth-ieee-float32
---ro one-way-available-bandwidth?
  |       rt-types:bandwidth-ieee-float32
---ro one-way-utilized-bandwidth?
  |       rt-types:bandwidth-ieee-float32
---ro two-way-delay?
  |       uint32
++--ro one-way-min-delay?  uint32
++--ro one-way-max-delay?  uint32
++--ro one-way-delay-variation?  uint32
++--ro one-way-packet-loss?  decimal64
++--ro two-way-min-delay?  uint32
++--ro two-way-max-delay?  uint32
++--ro two-way-delay-variation?  uint32
++--ro two-way-packet-loss?  decimal64
++--ro threshold-in
++--ro one-way-delay?  uint32
|++--ro one-way-residual-bandwidth?
|   rt-types:bandwidth-ieee-float32
|++--ro one-way-available-bandwidth?
|   rt-types:bandwidth-ieee-float32
|++--ro one-way-utilized-bandwidth?
|   rt-types:bandwidth-ieee-float32
++--ro two-way-delay?  uint32
++--ro one-way-min-delay?  uint32
++--ro one-way-max-delay?  uint32
++--ro one-way-delay-variation?  uint32
++--ro one-way-packet-loss?  decimal64
++--ro two-way-min-delay?  uint32
++--ro two-way-max-delay?  uint32
++--ro two-way-delay-variation?  uint32
++--ro two-way-packet-loss?  decimal64
++--ro threshold-accelerated-advertisement
++--ro one-way-delay?  uint32
|++--ro one-way-residual-bandwidth?
|   rt-types:bandwidth-ieee-float32
|++--ro one-way-available-bandwidth?
|   rt-types:bandwidth-ieee-float32
|++--ro one-way-utilized-bandwidth?
|   rt-types:bandwidth-ieee-float32
++--ro two-way-delay?  uint32
++--ro one-way-min-delay?  uint32
++--ro one-way-max-delay?  uint32
++--ro one-way-delay-variation?  uint32
++--ro one-way-packet-loss?  decimal64
++--ro two-way-min-delay?  uint32
++--ro two-way-max-delay?  uint32
++--ro two-way-delay-variation?  uint32
++--ro two-way-packet-loss?  decimal64
/tet:interface-switching-capability:
++--rw packet-switch-capable
++--rw minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
++--rw interface-mtu?  uint16
augment /nw:networks/nw:network/nt:link/tet:te
   /tet:te-link-attributes
   /tet:interface-switching-capability:
      ++--rw packet-switch-capable
      ++--rw minimum-lsp-bandwidth? rt-types:bandwidth-ieee-float32
      ++--rw interface-mtu? uint16
augment /nw:networks/nw:network/nt:link/tet:te
   /tet:information-source-entry
   /tet:interface-switching-capability:
      ++--ro packet-switch-capable
      ++--ro minimum-lsp-bandwidth? rt-types:bandwidth-ieee-float32
      ++--ro interface-mtu? uint16

5. YANG Modules

5.1. Layer 3 TE Topology Module

This module references [RFC8345], [RFC8346], and [I-D.ietf-teas-yang-te-topo].

<CODE BEGINS> file "ietf-l3-te-topology@2019-02-28.yang"
module ietf-l3-te-topology {
   yang-version 1.1;
   prefix "l3tet";

   import ietf-network {
      prefix "nw";
      reference "RFC 8345: A YANG Data Model for Network Topologies";
   }
import ietf-network-topology {
   prefix "nt";
   reference "RFC 8345: A YANG Data Model for Network Topologies";
}
import ietf-l3-unicast-topology {
   prefix "l3t";
   reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
}
import ietf-te-topology {
   prefix "tet";
   reference
      "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic Engineering (TE) Topologies";
}
organization
"IETF Traffic Engineering Architecture and Signaling (TEAS)
Working Group";

contact
"WG Web:  <http://tools.ietf.org/wg/teas/>
WG List:  <mailto:teas@ietf.org>
Editor:      Xufeng Liu
            <mailto:xufeng.liu.ietf@gmail.com>
Editor:      Igor Bryskin
            <mailto:Igor.Bryskin@huawei.com>
Editor:      Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
Editor:      Tarek Saad
            <mailto:tsaad@cisco.com>
Editor:      Himanshu Shah
            <mailto:hshah@ciena.com>
Editor:      Oscar Gonzalez De Dios
            <mailto:oscar.gonzalezdedios@telefonica.com>"

description
"YANG data model for representing and manipulating Layer 3 TE
Topologies.

Copyright (c) 2018 IETF Trust and the persons identified as
authors of the code. All rights reserved.

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

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

revision 2019-02-28 {

description "Initial revision";
reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";
}

grouping l3-te-topology-type {
  description "Identifies the L3 TE topology type.";
  container l3-te {
    presence "indicates L3 TE Topology";
    description "Its presence identifies the L3 TE topology type.";
  }
}

augment "/nw:networks/nw:network/nw:network-types/" + "l3t:l3-unicast-topology" {
  description "Defines the L3 TE topology type.";
  uses l3-te-topology-type;
}

augment "/nw:networks/nw:network/l3t:l3-topology-attributes" {
  when "../nw:network-types/l3t:l3-unicast-topology/l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment topology configuration";
  uses l3-te-topology-attributes;
}

augment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes" {
  when "../..//nw:network-types/l3t:l3-unicast-topology/" + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment node configuration";
  uses l3-te-node-attributes;
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/" + "l3t:l3-termination-point-attributes" {
  when "../..//nw:network-types/l3t:l3-unicast-topology/" + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment termination point configuration";
  uses l3-te-tp-attributes;
}

augment "/nw:networks/nw:network/nt:link/l3t:l3-link-attributes" {
  when "../..//nw:network-types/l3t:l3-unicast-topology/" + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
grouping l3-te-topology-attributes {
    description "L3 TE topology scope attributes";
    container l3-te-topology-attributes {
        must "/nw:networks/nw:network"
        + "/[nw:network-id = current()/network-ref]/nw:network-types/"
        + "tet:te-topology" {
            error-message
            "The referenced network must be a TE topology.";
            description
            "The referenced network must be a TE topology.";
        } // l3-te-topology-attributes
    } // l3-te-topology-attributes
}

grouping l3-te-node-attributes {
    description "L3 TE node scope attributes";
    container l3-te-node-attributes {
        must "/nw:networks/nw:network"
        + "/[nw:network-id = current()/network-ref]/nw:network-types/"
        + "tet:te-topology" {
            error-message
            "The referenced network must be a TE topology.";
            description
            "The referenced network must be a TE topology.";
        } // l3-te-node-attributes
    } // l3-te-node-attributes
}

grouping l3-te-tp-attributes {
    description "L3 TE termination point scope attributes";
    container l3-te-tp-attributes {
        must "/nw:networks/nw:network"
        + "/[nw:network-id = current()/network-ref]/nw:network-types/"
        + "tet:te-topology" {
            error-message
            "The referenced network must be a TE topology.";
            description
            "The referenced network must be a TE topology.";
        } // l3-te-tp-attributes
    } // l3-te-tp-attributes
}
description "Containing TE termination point references";
uses nt:tp-ref;
} // 13-te
} // 13-te-tp-attributes

grouping 13-te-link-attributes {
   description "L3 TE link scope attributes";
   container 13-te-link-attributes {
      must "/nw:networks/nw:network" + 
      "[nw:network-id = current()/network-ref]/nw:network-types/"
      + "tet:te-topology" {
         error-message
         "The referenced network must be a TE topology.";
         description
         "The referenced network must be a TE topology.";
      }
      description "Containing TE link references";
      uses nt:link-ref;
   }
} // 13-te-link-attributes

5.2. Packet Switching TE Topology Module

This module references [RFC7471], [RFC7810], [RFC7823], [RFC8294], [RFC8345], [RFC8346]. [I-D.ietf-teas-yang-te-types], and [I-D.ietf-teas-yang-te-topo].

<CODE BEGINS> file "ietf-te-topology-packet@2019-02-28.yang"
module ietf-te-topology-packet {
   yang-version 1.1;
   prefix "tet-pkt";

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

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

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

import ietf-te-topology {
    prefix "tet";
    reference
        "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic
         Engineering (TE) Topologies";
}

import ietf-te-types {
    prefix "te-types";
    reference
        "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG
         Types";
}

import ietf-te-packet-types {
    prefix "te-packet-types";
    reference
        "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG
         Types";
}

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

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

    Editor:   Xufeng Liu
              <mailto:xufeng.liu.ietf@gmail.com>

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

    Editor:   Vishnu Pavan Beeram
              <mailto:vbeeram@juniper.net>

    Editor:   Tarek Saad
              <mailto:tsaad@cisco.com>

    Editor:   Himanshu Shah
<mailto:hshah@ciena.com>

Editor: Oscar Gonzalez De Dios
<mailto:oscar.gonzalezdedios@telefonica.com>

description
"YANG data model for representing and manipulating PSC (Packet Switching) TE Topologies.

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

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

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

revision 2019-02-28 {
  description "Initial revision";
  reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";
}

/*
 * Features
 */

feature te-performance-metric {
  description
    "This feature indicates that the system supports TE performance metric.";
  reference
}

/*
 * Groupings
 */

grouping packet-switch-capable-container {
  description
The container of packet switch capable attributes.

container packet-switch-capable {
    description "Interface has packet-switching capabilities."
    leaf minimum-lsp-bandwidth {
        type rt-types:bandwidth-ieee-float32;
        description "Minimum LSP Bandwidth. Units in bytes per second"
    }
    leaf interface-mtu {
        type uint16;
        description "Interface MTU."
    }
}

/* Augmentations */
/* Augmentations to connectivity-matrix */
augment "/nw:networks/nw:network/nw:node/tet:te/" 
    + "tet:te-node-attributes/tet:connectivity-matrices" {
    description "Parameters for PSC TE topology."
    uses te-packet-types:performance-metrics-attributes-packet {
        if-feature te-performance-metric;
        refine performance-metrics-one-way {
            config false;
        }
        refine performance-metrics-two-way {
            config false;
        }
    }
    uses te-packet-types:performance-metrics-throttle-container-packet {
        if-feature te-performance-metric;
    }
}

augment "/nw:networks/nw:network/nw:node/tet:te/" 
    + "tet:te-node-attributes/tet:connectivity-matrices/" 
    + "tet:connectivity-matrix" {
    description "Parameters for PSC TE topology."
    uses te-packet-types:performance-metrics-attributes-packet {
        if-feature te-performance-metric;
        refine performance-metrics-one-way {
            config false;
        }
    }
}

Liu, et al. Expires September 12, 2019 [Page 34]
+ "tet:information-source-entry/tet:connectivity-matrices/" 
+ "tet:connectivity-matrix" { 
  description 
  "Parameters for PSC TE topology.";
  uses te-packet-types:performance-metrics-attributes-packet { 
    if-feature te-performance-metric;
  }
  uses 
  te-packet-types:performance-metrics-throttle-container-packet { 
    if-feature te-performance-metric;
  }
}

/* Augmentations to tunnel-termination-point */
augment "/nw:networks/nw:network/nw:node/tet:te/" 
  + "tet:tunnel-termination-point/" 
  + "tet:local-link-connectivities" { 
  description 
  "Parameters for PSC TE topology.";
  uses te-packet-types:performance-metrics-attributes-packet { 
    if-feature te-performance-metric;
    refine performance-metrics-one-way { 
      config false;
    }
  }
}
config false;
}
refine performance-metrics-two-way {
    config false;
}
}
uses
te-packet-types:performance-metrics-throttle-container-packet {
    if-feature te-performance-metric;
}
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:tunnel-termination-point/"
    + "tet:local-link-connectivities/"
    + "tet:local-link-connectivity" {
    description
    "Parameters for PSC TE topology."
    uses te-packet-types:performance-metrics-attributes-packet {
        if-feature te-performance-metric;
        refine performance-metrics-one-way {
            config false;
        }
        refine performance-metrics-two-way {
            config false;
        }
    }
    uses
te-packet-types:performance-metrics-throttle-container-packet {
        if-feature te-performance-metric;
    }
}

/* Augmentations to te-link-attributes */
augment "/nw:networks/tet:te/tet:templates/"
    + "tet:link-template/tet:te-link-attributes" {
    when "tet:interface-switching-capability "
        + "[tet:switching-capability = 'te-types:switching-pscl']"
    {
    description "Valid only for PSC";
    }
    description
    "Parameters for PSC TE topology."
    uses te-packet-types:performance-metrics-attributes-packet {
        if-feature te-performance-metric;
        refine performance-metrics-one-way {
            config false;
        }
        refine performance-metrics-two-way {

config false;
}
}
uses
te-packet-types:performance-metrics-throttle-container-packet {
  if-feature te-performance-metric;
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:te-link-attributes" {
    when "tet:interface-switching-capability"
      + "[tet:switching-capability = 'te-types:switching-psc1']"
      {
        description "Valid only for PSC";
      }
    description
      "Parameters for PSC TE topology.";
    uses te-packet-types:performance-metrics-attributes-packet {
      if-feature te-performance-metric;
      refine performance-metrics-one-way {
        config false;
      }
      refine performance-metrics-two-way {
        config false;
      }
    }
    uses
      te-packet-types:performance-metrics-throttle-container-packet {
        if-feature te-performance-metric;
      }
  }

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:information-source-entry" {
    when "tet:interface-switching-capability"
      + "[tet:switching-capability = 'te-types:switching-psc1']"
      {
        description "Valid only for PSC";
      }
    description
      "Parameters for PSC TE topology.";
    uses te-packet-types:performance-metrics-attributes-packet {
      if-feature te-performance-metric;
    }
    uses
      te-packet-types:performance-metrics-throttle-container-packet {
        if-feature te-performance-metric;
      }
  }
/* Augmentations to interface-switching-capability */
   + "tet:link-template/tet:te-link-attributes/
     + tet:interface-switching-capability" {
   when "tet:switching-capability = 'te-types:switching-pscl'" {
     description "Valid only for PSC";
   }
   description
     "Parameters for PSC TE topology."
   uses packet-switch-capable-container;
}

   + "tet:te-link-attributes/
     + tet:interface-switching-capability" {
   when "tet:switching-capability = 'te-types:switching-pscl'" {
     description "Valid only for PSC";
   }
   description
     "Parameters for PSC TE topology."
   uses packet-switch-capable-container;
}

   + "tet:information-source-entry/
     + tet:interface-switching-capability" {
   when "tet:switching-capability = 'te-types:switching-pscl'" {
     description "Valid only for PSC";
   }
   description
     "Parameters for PSC TE topology."
   uses packet-switch-capable-container;
}

<CODE ENDS>

6. IANA Considerations

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

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

Liu, et al. Expires September 12, 2019 [Page 38]
This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

```
name:         ietf-l3-te-topology
prefix:       l3te
reference:    RFC XXXX
```

```
name:         ietf-l3-te-topology-state
prefix:       l3te-s
reference:    RFC XXXX
```

```
name:         ietf-te-topology-packet
prefix:       tet-pkt
reference:    RFC XXXX
```
7. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that 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., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

```
l3-te
    This subtree specifies the layer 3 TE topology type. Modifying the configurations can make layer 3 TE topology type invalid and cause interruption to all layer 3 TE networks.
/fw:networks/fw:network/l3t:l3-topology-attributes/l3-te-topology-
attributes
    This subtree specifies the topology-wide configurations, including the reference to a TE topology from a layer 3 network topology. Modifying the configurations here can cause traffic disabled or rerouted in this topology and the connected topologies.
/fw:networks/fw:network/fw:node/l3t:l3-node-attributes/l3-te-node-
attributes
    This subtree specifies the configurations of layer 3 TE nodes. Modifying the configurations in this subtree can change the relationship between a TE node and a layer 3 node, causing traffic
```
disabled or rerouted in the specified nodes and the related layer 3 topologies.

```
```

This subtree specifies the configurations of layer 3 TE link termination points. Modifying the configurations in this subtree can change the relationship between a TE link termination point and a layer 3 link termination point, causing traffic disabled or rerouted on the related layer 3 links and the related layer 3 topologies.

```
/nw:networks/nw:network/nt:link/l3t:l3-link-attributes/l3-te-link-attributes
```

This subtree specifies the configurations of layer 3 TE links. Modifying the configurations in this subtree can change the relationship between a TE link and a layer 3 link, causing traffic disabled or rerouted on the specified layer 3 link and the related layer 3 topologies.

**performance-metric containers**

The container "performance-metric" is augmented to multiple locations of the base TE topology model, as specified in Section 3.2. Modifying the configuration in such a container can change the behaviours of performance metric monitoring, causing traffic disabled or rerouted on the related layer 3 links, nodes, or topologies.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

```
/nw:networks/nw:network/nw:network-types/l3t:l3-unicast-topology/l3-te
```

Unauthorized access to this subtree can disclose the layer 3 TE topology type.

```
/nw:networks/nw:network/l3t:l3-topology-attributes/l3-te-topology-attributes
```

Unauthorized access to this subtree can disclose the topology-wide configurations, including the reference to a TE topology from a layer 3 network topology.

```
/nw:networks/nw:network/l3t:l3-node-attributes/l3-te-node-attributes
```

Unauthorized access to this subtree can disclose the layer 3 node attributes and their sensitivity/vulnerability.
Unauthorized access to this subtree can disclose the operational state information of layer 3 TE nodes.

/w/nw:networks/nw:network/nw:node/nt:termination-point/l3t:l3-termination-point-attributes/l3-te-tp-attributes
Unauthorized access to this subtree can disclose the operational state information of layer 3 TE link termination points.

/w/nw:networks/nw:network/nt:link/l3t:l3-link-attributes/l3-te-link-attributes
Unauthorized access to this subtree can disclose the operational state information of layer 3 TE links.

performance-metric containers
The container "performance-metric" is augmented to multiple locations of the base TE topology model, as specified in Section 3.2. Unauthorized access to this subtree can disclose the operational state information of performance metric monitoring.

8. References

8.1. Normative References


8.2. Informative References


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

The YANG modules ietf-l3-te-topology and ietf-te-topology-packet defined in this document are designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342].  In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion modules, ietf-l3-te-topology-state and ietf-te-topology-packet-state, are defined as state models, which mirror the modules ietf-l3-te-topology and ietf-te-topology-packet defined earlier in this document.  However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion modules, ietf-l3-te-topology-state and ietf-te-topology-packet-state, are redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion modules mirrors that of the corresponding NMDA models, the YANG trees of the companion modules are not depicted separately.

A.1.  Layer 3 TE Topology State Module

This module references [RFC8345], and [RFC8346].

<CODE BEGINS> file "ietf-l3-te-topology-state@2019-02-28.yang"
module ietf-l3-te-topology-state {
  yang-version 1.1;
  prefix "l3tet-s";

  import ietf-l3-te-topology {
    prefix "l3tet";
  }
  import ietf-network-state {
    prefix "nw-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-network-topology-state {
    prefix "nt-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }
}

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

contact
"WG Web: <http://tools.ietf.org/wg/teas/>
WG List: <mailto:teas@ietf.org>
Editor: Xufeng Liu
<mailto:xufeng.liu.ietf@gmail.com>
Editor: Igor Bryskin
<mailto:Igor.Bryskin@huawei.com>
Editor: Vishnu Pavan Beeram
<mailto:vbeeram@juniper.net>
Editor: Tarek Saad
<mailto:tsaad@cisco.com>
Editor: Himanshu Shah
<mailto:hshah@ciena.com>
Editor: Oscar Gonzalez De Dios
<mailto:oscar.gonzalezdedios@telefonica.com>"

description
"YANG data model for representing operational state information
of Layer 3 TE Topologies, when NMDA is not supported.

Copyright (c) 2018 IETF Trust and the persons identified as
authors of the code. All rights reserved.

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

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

revision 2019-02-28 {

description "Initial revision";
reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";
}

augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
  + "l3t-s:l3-unicast-topology" {
    description
      "Defines the L3 TE topology type."
    uses l3tet:l3-te-topology-type;
  }

augment "/nw-s:networks/nw-s:network/"
  + "l3t-s:l3-topology-attributes" {
    when "/nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "l3tet-s:l3-te" {
        description "Augment only for L3 TE topology";
      }
    description "Augment topology configuration"
    uses l3tet:l3-te-topology-attributes;
  }

augment "/nw-s:networks/nw-s:network/nw-s:node/"
  + "l3t-s:l3-node-attributes" {
    when "/nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "l3tet-s:l3-te" {
        description "Augment only for L3 TE topology";
      }
    description "Augment node configuration"
    uses l3tet:l3-te-node-attributes;
  }

augment "/nw-s:networks/nw-s:network/nw-s:node/nt-s:termination-point/"
  + "l3t-s:l3-termination-point-attributes" {
    when "/nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "l3tet-s:l3-te" {
        description "Augment only for L3 TE topology";
      }
    description "Augment termination point configuration"
    uses l3tet:l3-te-tp-attributes;
  }

augment "/nw-s:networks/nw-s:network/nt-s:link/"
  + "l3t-s:l3-link-attributes" {
    when "/nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "l3tet-s:l3-te" {
        description "Augment only for L3 TE topology";
      }
    description "Augment link configuration"
    uses l3tet:l3-te-link-attributes;
  }

A.2. Packet Switching TE Topology State Module

<CODE BEGINS> file "ietf-te-topology-packet-state@2019-02-28.yang"
module ietf-te-topology-packet-state {
  yang-version 1.1;

  prefix "tet-pkt-s";

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

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

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

  import ietf-te-topology-state {
    prefix "tet-s";
    reference "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic Engineering (TE) Topologies";
  }

  import ietf-te-types {
    prefix "te-types";
    reference "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
  }

  import ietf-te-packet-types {
    prefix "te-packet-types";
    reference "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
  }
}
organization
    "Traffic Engineering Architecture and Signaling (TEAS)
    Working Group";

contact
    "WG Web: <http://tools.ietf.org/wg/teas/>
    WG List: <mailto:teas@ietf.org>
    Editor: Xufeng Liu
            <mailto:xufeng.liu.ietf@gmail.com>
    Editor: Igor Bryskin
            <mailto:Igor.Bryskin@huawei.com>
    Editor: Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
    Editor: Tarek Saad
            <mailto:tsaad@cisco.com>
    Editor: Himanshu Shah
            <mailto:hshah@ciena.com>
    Editor: Oscar Gonzalez De Dios
            <mailto:oscar.gonzalezdedios@telefonica.com>";

description
    "YANG data model for representing operational state information
    of PSC (Packet Switching) TE Topologies, when NMDA is not
    supported.

    Copyright (c) 2018 IETF Trust and the persons identified as
    authors of the code. All rights reserved.

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

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

revision 2019-02-28 {
    description "Initial revision";
reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";
}

/*
* Augmentations
*/
/* Augmentations to connectivity-matrix */
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/
 + "tet-s:te-node-attributes/tet-s:connectivity-matrices" {
  description
  "Parameters for PSC (Packet Switching) TE topology.";
  uses te-packet-types:performance-metrics-attributes-packet {
    if-feature tet-pkt:te-performance-metric;
  }
  uses
    te-packet-types:performance-metrics-throttle-container-packet {
      if-feature tet-pkt:te-performance-metric;
    }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/
 + "tet-s:te-node-attributes/tet-s:connectivity-matrices/
 + "tet-s:connectivity-matrix" {
  description
  "Parameters for PSC TE topology.";
  uses te-packet-types:performance-metrics-attributes-packet {
    if-feature tet-pkt:te-performance-metric;
  }
  uses
    te-packet-types:performance-metrics-throttle-container-packet {
      if-feature tet-pkt:te-performance-metric;
    }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/
 + "tet-s:information-source-entry/
 + "tet-s:connectivity-matrices" {
  description
  "Parameters for PSC TE topology.";
  uses te-packet-types:performance-metrics-attributes-packet {
    if-feature tet-pkt:te-performance-metric;
  }
  uses
    te-packet-types:performance-metrics-throttle-container-packet {
      if-feature tet-pkt:te-performance-metric;
    }
}
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
   + "tet-s:information-source-entry/"
   + "tet-s:connectivity-matrices/
   + "tet-s:connectivity-matrix" {
   description
   "Parameters for PSC TE topology."
   uses te-packet-types:performance-metrics-attributes-packet {
      if-feature tet-pkt:te-performance-metric;
   }
   uses
   te-packet-types:performance-metrics-throttle-container-packet {
      if-feature tet-pkt:te-performance-metric;
   }
}
/* Augmentations to tunnel-termination-point */
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
   + "tet-s:tunnel-termination-point/"
   + "tet-s:local-link-connectivities" {
   description
   "Parameters for PSC TE topology."
   uses te-packet-types:performance-metrics-attributes-packet {
      if-feature tet-pkt:te-performance-metric;
   }
   uses
   te-packet-types:performance-metrics-throttle-container-packet {
      if-feature tet-pkt:te-performance-metric;
   }
}
/* Augmentations to te-link-attributes */
augment "/nw-s:networks/tet-s:te/tet-s:templates/
   + "tet-s:link-template/tet-s:te-link-attributes" {
when "tet-s:interface-switching-capability "
  + "[tet-s:switching-capability = ‘te-types:switching-psc1’]" { 
    description "Valid only for PSC";
  }
}
description
 "Parameters for PSC TE topology.";
uses te-packet-types:performance-metrics-attributes-packet {
  if-feature tet-pkt:te-performance-metric;
}
uses
  te-packet-types:performance-metrics-throttle-container-packet {
    if-feature tet-pkt:te-performance-metric;
  }
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
  + "tet-s:te-link-attributes" {
  when "tet-s:interface-switching-capability"
    + "[tet-s:switching-capability = ‘te-types:switching-psc1’]" { 
      description "Valid only for PSC";
    }
  }
description
 "Parameters for PSC TE topology.";
uses te-packet-types:performance-metrics-attributes-packet {
  if-feature tet-pkt:te-performance-metric;
}
uses
  te-packet-types:performance-metrics-throttle-container-packet {
    if-feature tet-pkt:te-performance-metric;
  }
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
  + "tet-s:information-source-entry" {
  when "tet-s:interface-switching-capability"
    + "[tet-s:switching-capability = ‘te-types:switching-psc1’]" { 
      description "Valid only for PSC";
    }
  }
description
 "Parameters for PSC TE topology.";
uses te-packet-types:performance-metrics-attributes-packet {
  if-feature tet-pkt:te-performance-metric;
}
uses
  te-packet-types:performance-metrics-throttle-container-packet {
    if-feature tet-pkt:te-performance-metric;
  }
}
/* Augmentations to interface-switching-capability */
augment "/nw-s:networks/tet-s:te/tet-s:templates/"
  + "tet-s:link-template/tet-s:te-link-attributes/
  + "tet-s:interface-switching-capability" {
  when "tet-s:switching-capability = 'te-types:switching-pscl' " {
    description "Valid only for PSC";
  }
  description
    "Parameters for PSC TE topology.";
  uses tet-pkt:packet-switch-capable-container;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
  + "tet-s:te-link-attributes/
  + "tet-s:interface-switching-capability" {
  when "tet-s:switching-capability = 'te-types:switching-pscl' " {
    description "Valid only for PSC";
  }
  description
    "Parameters for PSC TE topology.";
  uses tet-pkt:packet-switch-capable-container;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
  + "tet-s:information-source-entry/
  + "tet-s:interface-switching-capability" {
  when "tet-s:switching-capability = 'te-types:switching-pscl' " {
    description "Valid only for PSC";
  }
  description
    "Parameters for PSC TE topology.";
  uses tet-pkt:packet-switch-capable-container;
}

<CODE ENDS>

Appendix B. Data Tree Example

This section contains an example of an instance data tree in the JSON encoding [RFC7951]. The example instantiates "ietf-l3-te-topology" for the topology that is depicted in the following diagram.
The corresponding instance data tree is depicted below. Note that some lines have been wrapped to adhere to the 72-character line limitation of RFCs.

```json
{
"ietf-network:networks": {
  "network": [
    {
      "network-types": {
        "ietf-te-topology:te-topology": {}
      },
      "network-id": "example-topo-te",
      "ietf-te-topology:provider-id": 200,
      "ietf-te-topology:client-id": 300,
      "ietf-te-topology:te-topology-id": "example-topo-te",
      "ietf-te-topology:te": {
      },
      "node": [
        {
          "node-id": "D1",
          "ietf-te-topology:te-node-id": "2.0.1.1",
          "ietf-te-topology:te": {
            "te-node-attributes": {
            }
          },
          "ietf-network-topology:termination-point": [
```
{ "tp-id": "1-2-1",
"ietf-te-topology:te-tp-id": 10201,
"ietf-te-topology:te": {
   "interface-switching-capability": [
   { "switching-capability": "switching-psc1",
     "encoding": "lsp-encoding-ethernet"
   }
   ]
},

{ "tp-id": "1-3-1",
"ietf-te-topology:te-tp-id": 10301,
"ietf-te-topology:te": {
   "interface-switching-capability": [
   { "switching-capability": "switching-psc1",
     "encoding": "lsp-encoding-ethernet"
   }
   ]
}
},

{ "node-id": "D2",
"ietf-te-topology:te-node-id": "2.0.2.1",
"ietf-te-topology:te": {
   "te-node-attributes": {
   }
},
"ietf-network-topology:termination-point": [
{ "tp-id": "2-1-1",
"ietf-te-topology:te-tp-id": 20101,
"ietf-te-topology:te": {
   "interface-switching-capability": [
   { "switching-capability": "switching-psc1",
     "encoding": "lsp-encoding-ethernet"
   }
   ]
}
]
},
{
    "node-id": "D3",
    "ietf-te-topology:te-node-id": "2.0.3.1",
    "ietf-te-topology:te": {
        "te-node-attributes": {
            ...
        }
    },
    "ietf-network-topology:termination-point": {
        "tp-id": "3-1-1",
        "ietf-te-topology:te-tp-id": 30101,
        "ietf-te-topology:te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-pscl",
                    "encoding": "lsp-encoding-ethernet"
                }
            ],
            "max-link-bandwidth": {
                "te-bandwidth": {"generic": "0x1p+18"
            }
        },
        "te-default-metric": 100
    }
}
}
"link-id": "D2,2-1-1,D1,1-2-1",
"source": {
  "source-node": "D2",
  "source-tp": "2-1-1"
},
"destination": {
  "dest-node": "D1",
  "dest-tp": "1-2-1"
},
"ietf-te-topology:te": {
  "te-link-attributes": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-pscl",
        "encoding": "lsp-encoding-ethernet"
      }
    ],
    "max-link-bandwidth": {
      "te-bandwidth": {
        "generic": "0x1p+18"
      }
    },
    "te-default-metric": 100
  }
},
"link-id": "D1,1-3-1,D3,3-1-1",
"source": {
  "source-node": "D1",
  "source-tp": "1-3-1"
},
"destination": {
  "dest-node": "D3",
  "dest-tp": "3-1-1"
},
"ietf-te-topology:te": {
  "te-link-attributes": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-pscl",
        "encoding": "lsp-encoding-ethernet"
      }
    ]
  }
}
"max-link-bandwidth": {
  "te-bandwidth": {
    "generic": "0x1p+18"
  },
  "te-default-metric": 100
},

"link-id": "D3,3-1-1,D1,1-3-1",
"source": {
  "source-node": "D3",
  "source-tp": "3-1-1"
},
"destination": {
  "dest-node": "D1",
  "dest-tp": "1-3-1"
},
"ietf-te-topology:te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-pscl",
      "encoding": "lsp-encoding-ethernet"
    }
  ],
  "max-link-bandwidth": {
    "te-bandwidth": {
      "generic": "0x1p+18"
    }
  },
  "te-default-metric": 100
}
"node": [
  {
    "node-id": "D1",
    "ietf-network-topology:termination-point": {
      "tp-id": "1-0-1",
      "ietf-l3-unicast-topology:l3-termination-point-attributes": {
        "unnumbered-id": 101
      }
    },
    "tp-id": "1-2-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {
      "unnumbered-id": 121,
      "ietf-l3-te-topology:l3-te-tp-attributes": {
        "tp-ref": "1-2-1"
      }
    },
    "tp-id": "1-3-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {
      "unnumbered-id": 131,
      "ietf-l3-te-topology:l3-te-tp-attributes": {
        "tp-ref": "1-3-1"
      }
    }
  },
  {
    "node-id": "D2",
    "ietf-network-topology:termination-point": {
      "tp-id": "2-0-1",
      "unnumbered-id": 201
    }
  }
]
"ietf-l3-unicast-topology:l3-termination-point-attributes": {  
  "unnumbered-id": 201
},

"tp-id": "2-1-1",
"ietf-l3-unicast-topology:l3-termination-point-attributes": {  
  "unnumbered-id": 211,
  "ietf-l3-te-topology:l3-te-tp-attributes": {  
    "tp-ref": "2-1-1"
  }
},

"tp-id": "2-3-1",
"ietf-l3-unicast-topology:l3-termination-point-attributes": {  
  "unnumbered-id": 231
}
},

"ietf-l3-unicast-topology:l3-node-attributes": {  
  "router-id": ["203.0.113.2"],
  "prefix": [
    {  
      "prefix": "203.0.113.2/32"
    }
  ],
  "ietf-l3-te-topology:l3-te-node-attributes": {  
    "node-ref": "D2"
  }
}
},

"node-id": "D3",
"ietf-network-topology:termination-point": [
  {  
    "tp-id": "3-1-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {  
      "unnumbered-id": 311,
      "ietf-l3-te-topology:l3-te-tp-attributes": {  
        "tp-ref": "3-1-1"
      }
    }
  },
  {  
    "tp-id": "3-2-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {  
      "unnumbered-id": 321
    }
  }
]
"ietf-l3-unicast-topology:l3-node-attributes": {
  "router-id": ["203.0.113.3"],
  "prefix": [
    {
      "prefix": "203.0.113.3/32"
    }
  ],
"ietf-l3-te-topology:l3-te-node-attributes": {
  "node-ref": "D3"
}
},
"ietf-network-topology:link": [
  {
    "link-id": "D1,1-2-1,D2,2-1-1",
    "source": {
      "source-node": "D1",
      "source-tp": "1-2-1"
    },
    "destination": {
      "dest-node": "D2",
      "dest-tp": "2-1-1"
    },
    "ietf-l3-unicast-topology:l3-link-attributes": {
      "metric1": "100",
      "ietf-l3-te-topology:l3-te-link-attributes": {
        "link-ref": "D1,1-2-1,D2,2-1-1"
      }
    }
  },
  {
    "link-id": "D2,2-1-1,D1,1-2-1",
    "source": {
      "source-node": "D2",
      "source-tp": "2-1-1"
    },
    "destination": {
      "dest-node": "D1",
      "dest-tp": "1-2-1"
    },
    "ietf-l3-unicast-topology:l3-link-attributes": {
      "metric1": "100",
      "ietf-l3-te-topology:l3-te-link-attributes": {
        "link-ref": "D2,2-1-1,D1,1-2-1"
      }
    }
  }
]
"link-id": "D3,3-1-1,D1,1-3-1",
"source": {
  "source-node": "D3",
  "source-tp": "3-1-1"
},
"destination": {
  "dest-node": "D1",
  "dest-tp": "1-3-1"
},
"ietf-l3-unicast-topology:13-link-attributes": {
  "metric1": "100",
  "ietf-l3-te-topology:13-te-link-attributes": {
    "link-ref": "D3,3-1-1,D1,1-3-1"
  }
}
},
{
  "link-id": "D1,1-3-1,D3,3-1-1",
  "source": {
    "source-node": "D1",
    "source-tp": "1-3-1"
  },
  "destination": {
    "dest-node": "D3",
    "dest-tp": "3-1-1"
  },
  "ietf-l3-unicast-topology:13-link-attributes": {
    "metric1": "100",
    "ietf-l3-te-topology:13-te-link-attributes": {
      "link-ref": "D1,1-3-1,D3,3-1-1"
    }
  }
},
{
  "link-id": "D2,2-3-1,D3,3-2-1",
  "source": {
    "source-node": "D2",
    "source-tp": "2-3-1"
  },
  "destination": {
    "dest-node": "D3",
    "dest-tp": "3-2-1"
  },
  "ietf-l3-unicast-topology:13-link-attributes": {
    "metric1": "100",
    "ietf-l3-te-topology:13-te-link-attributes": {
      "link-ref": "D2,2-3-1,D3,3-2-1"
    }
  }
}
"metric1": "100"
}
},
{
"link-id": "D3,3-2-1,D2,2-3-1",
"source": {
"source-node": "D3",
"source-tp": "3-2-1"
},
"destination": {
"dest-node": "D2",
"dest-tp": "2-3-1"
},
"ietf-l3-unicast-topology:l3-link-attributes": {
"metric1": "100"
}
}
}
}
}

Authors’ Addresses

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

Igor Bryskin
Huawei Technologies
EMail: Igor.Bryskin@huawei.com

Vishnu Pavan Beeram
Juniper Networks
EMail: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc
EMail: tsaad@cisco.com

Yang model for requesting Path Computation
draft-ietf-teas-yang-path-computation-05.txt

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html
Abstract

There are scenarios, typically in a hierarchical SDN context, where the topology information provided by a TE network provider may not be sufficient for its client to perform end-to-end path computation. In these cases the client would need to request the provider to calculate some (partial) feasible paths.

This document defines a YANG data model for a stateless RPC to request path computation. This model complements the stateful solution defined in [TE-TUNNEL].

Moreover this document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

Table of Contents

1. Introduction...................................................3
   1.1. Terminology...............................................5
2. Use Cases......................................................5
   2.1. Packet/Optical Integration................................5
   2.2. Multi-domain TE Networks................................10
   2.3. Data center interconnections............................12
3. Motivations...................................................14
   3.1. Motivation for a YANG Model.............................14
   3.1.1. Benefits of common data models.......................14
3.1.2. Benefits of a single interface.......................15
3.1.3. Extensibility........................................15
3.2. Interactions with TE Topology............................16
  3.2.1. TE Topology Aggregation..............................17
  3.2.2. TE Topology Abstraction..............................20
  3.2.3. Complementary use of TE topology and path computation21
3.3. Stateless and Stateful Path Computation..................24
  3.3.1. Temporary reporting of the computed path state....26
4. Path Computation and Optimization for multiple paths......28
5. YANG Model for requesting Path Computation...............29
  5.1. Synchronization of multiple path computation requests...29
  5.2. Returned metric values................................31
6. YANG model for stateless TE path computation..............33
  6.1. YANG Tree...............................................33
  6.2. YANG Module............................................43
7. Security Considerations.....................................58
8. IANA Considerations.........................................59
9. References................................................59
  9.1. Normative References...................................59
  9.1. Informative References................................60
10. Acknowledgments.............................................61
Appendix A. Examples of dimensioning the "detailed connectivity
  matrix" 62

1. Introduction

There are scenarios, typically in a hierarchical SDN context, where
the topology information provided by a TE network provider may not
be sufficient for its client to perform end-to-end path computation.
In these cases the client would need to request the provider to
calculate some (partial) feasible paths, complementing his topology
knowledge, to make his end-to-end path computation feasible.

This type of scenarios can be applied to different interfaces in
different reference architectures:

- ABNO control interface [RFC7491], in which an Application Service
  Coordinator can request ABNO controller to take in charge path
  calculation (see Figure 1 in [RFC7491]).
A controller hierarchy is defined, the need for path computation arises on both interfaces CMI (interface between Customer Network Controller (CNC) and Multi Domain Service Coordinator (MDSC)) and/or MPI (interface between MSDC-PNC). [RFC8454] describes an information model for the Path Computation request.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).

Path Computation Elements, Controllers and Orchestrators perform their operations based on Traffic Engineering Databases (TED). Such TEDs can be described, in a technology agnostic way, with the YANG Data Model for TE Topologies [TE-TOPO]. Furthermore, the technology specific details of the TED are modeled in the augmented TE topology models (e.g. [OTN-TOPO] for OTN ODU technologies).

The availability of such topology models allows providing the TED using YANG-based protocols (e.g., NETCONF or RESTCONF). Furthermore, it enables a PCE/Controller performing the necessary abstractions or modifications and offering this customized topology to another PCE/Controller or high level orchestrator.

Note: This document assumes that the client of the YANG data model defined in this document may not implement a "PCE" functionality, as defined in [RFC4655].

The tunnels that can be provided over the networks described with the topology models can be also set-up, deleted and modified via YANG-based protocols (e.g., NETCONF or RESTCONF) using the TE-Tunnel Yang model [TE-TUNNEL].

This document proposes a YANG model for a path computation request defined as a stateless RPC, which complements the stateful solution defined in [TE-TUNNEL].

Moreover, this document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.
1.1. Terminology

TED: The traffic engineering database is a collection of all TE information about all TE nodes and TE links in a given network.

PCE: A Path Computation Element (PCE) is an entity that is capable of computing a network path or route based on a network graph, and of applying computational constraints during the computation. The PCE entity is an application that can be located within a network node or component, on an out-of-network server, etc. For example, a PCE would be able to compute the path of a TE LSP by operating on the TED and considering bandwidth and other constraints applicable to the TE LSP service request. [RFC4655]

2. Use Cases

This section presents different use cases, where a client needs to request underlying SDN controllers for path computation.

The presented uses cases have been grouped, depending on the different underlying topologies: a) Packet-Optical integration; b) Multi-domain Traffic Engineered (TE) Networks; and c) Data center interconnections.

2.1. Packet/Optical Integration

In this use case, an Optical network is used to provide connectivity to some nodes of a Packet network (see Figure 1).
Figure 1 - Packet/Optical Integration Use Case
Figure 1 as well as Figure 2 below only show a partial view of the packet network connectivity, before additional packet connectivity is provided by the Optical network.

It is assumed that the Optical network controller provides to the packet/optical coordinator an abstracted view of the Optical network. A possible abstraction could be to represent the whole optical network as one "virtual node" with "virtual ports" connected to the access links, as shown in Figure 2.

It is also assumed that Packet network controller can provide the packet/optical coordinator the information it needs to setup connectivity between packet nodes through the Optical network (e.g., the access links).

The path computation request helps the coordinator to know the real connections that can be provided by the optical network.
In this use case, the coordinator needs to setup an optimal underlying path for an IP link between R1 and R2.

As depicted in Figure 2, the coordinator has only an "abstracted view" of the physical network, and it does not know the feasibility or the cost of the possible optical paths (e.g., VP1-VP4 and VP2-VP5), which depend from the current status of the physical resources within the optical network and on vendor-specific optical attributes.

The coordinator can request the underlying Optical domain controller to compute a set of potential optimal paths, taking into account optical constraints. Then, based on its own constraints, policy and knowledge (e.g. cost of the access links), it can choose which one of these potential paths to use to setup the optimal end-to-end path crossing optical network.
Figure 3 - Packet/Optical Path Computation Example

For example, in Figure 3, the Coordinator can request the Optical network controller to compute the paths between VP1-VP4 and VP2-VP5 and then decide to setup the optimal end-to-end path using the VP2-VP5 Optical path even this is not the optimal path from the Optical domain perspective.

Considering the dynamicity of the connectivity constraints of an Optical domain, it is possible that a path computed by the Optical network controller when requested by the Coordinator is no longer valid/available when the Coordinator requests it to be setup up. This is further discussed in section 3.3.

2.2. Multi-domain TE Networks

In this use case there are two TE domains which are interconnected together by multiple inter-domains links.

A possible example could be a multi-domain optical network.
In order to setup an end-to-end multi-domain TE path (e.g., between nodes A and H), the multi-domain controller needs to know the feasibility or the cost of the possible TE paths within the two TE domains, which depend from the current status of the physical resources within each TE network. This is more challenging in case of optical networks because the optimal paths depend also on vendor-
specific optical attributes (which may be different in the two domains if they are provided by different vendors).

In order to setup a multi-domain TE path (e.g., between nodes A and H), the multi-domain controller can request the TE domain controllers to compute a set of intra-domain optimal paths and take decisions based on the information received. For example:

- The multi-domain controller asks TE domain controllers to provide set of paths between A-C, A-D, E-H and F-H
- TE domain controllers return a set of feasible paths with the associated costs: the path A-C is not part of this set (in optical networks, it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller)
- The multi-domain controller will select the path A-D-F-H since it is the only feasible multi-domain path and then request the TE domain controllers to setup the A-D and F-H intra-domain paths
- If there are multiple feasible paths, the multi-domain controller can select the optimal path knowing the cost of the intra-domain paths (provided by the TE domain controllers) and the cost of the inter-domain links (known by the multi-domain controller)

This approach may have some scalability issues when the number of TE domains is quite big (e.g. 20).

In this case, it would be worthwhile using the abstract TE topology information provided by the TE domain controllers to limit the number of potential optimal end-to-end paths and then request path computation to fewer TE domain controllers in order to decide what the optimal path within this limited set is.

For more details, see section 3.2.3.

2.3. Data center interconnections

In these use case, there is a TE domain which is used to provide connectivity between data centers which are connected with the TE domain using access links.
Figure 5 - Data Center Interconnection Use Case

In this use case, there is need to transfer data from Data Center 1 (DC1) to either DC2 or DC3 (e.g. workload migration).

The optimal decision depends both on the cost of the TE path (DC1-DC2 or DC1-DC3) and of the data center resources within DC2 or DC3.

The cloud network orchestrator needs to make a decision for optimal connection based on TE Network constraints and data centers.
resources. It may not be able to make this decision because it has only an abstract view of the TE network (as in use case in 2.1).

The cloud network orchestrator can request to the TE network controller to compute the cost of the possible TE paths (e.g., DC1-DC2 and DC1-DC3) and to the DC controller to provide the information it needs about the required data center resources within DC2 and DC3 and then it can take the decision about the optimal solution based on this information and its policy.

3. Motivations

This section provides the motivation for the YANG model defined in this document.

Section 3.1 describes the motivation for a YANG model to request path computation.

Section 3.2 describes the motivation for a YANG model which complements the TE Topology YANG model defined in [TE-TOPO].

Section 3.3 describes the motivation for a stateless YANG RPC which complements the TE Tunnel YANG model defined in [TE-TUNNEL].

3.1. Motivation for a YANG Model

3.1.1. Benefits of common data models

The YANG data model for requesting path computation is closely aligned with the YANG data models that provide (abstract) TE topology information, i.e., [TE-TOPO] as well as that are used to configure and manage TE Tunnels, i.e., [TE-TUNNEL].

There are many benefits in aligning the data model used for path computation requests with the YANG data models used for TE topology information and for TE Tunnels configuration and management:

- There is no need for an error-prone mapping or correlation of information.
- It is possible to use the same endpoint identifiers in path computation requests and in the topology modeling.
- The attributes used for path computation constraints are the same as those used when setting up a TE Tunnel.
3.1.2. Benefits of a single interface

The system integration effort is typically lower if a single, consistent interface is used by controllers, i.e., one data modeling language (i.e., YANG) and a common protocol (e.g., NETCONF or RESTCONF).

Practical benefits of using a single, consistent interface include:

1. Simple authentication and authorization: The interface between different components has to be secured. If different protocols have different security mechanisms, ensuring a common access control model may result in overhead. For instance, there may be a need to deal with different security mechanisms, e.g., different credentials or keys. This can result in increased integration effort.

2. Consistency: Keeping data consistent over multiple different interfaces or protocols is not trivial. For instance, the sequence of actions can matter in certain use cases, or transaction semantics could be desired. While ensuring consistency within one protocol can already be challenging, it is typically cumbersome to achieve that across different protocols.

3. Testing: System integration requires comprehensive testing, including corner cases. The more different technologies are involved, the more difficult it is to run comprehensive test cases and ensure proper integration.

4. Middle-box friendliness: Provider and consumer of path computation requests may be located in different networks, and middle-boxes such as firewalls, NATs, or load balancers may be deployed. In such environments it is simpler to deploy a single protocol. Also, it may be easier to debug connectivity problems.

5. Tooling reuse: Implementers may want to implement path computation requests with tools and libraries that already exist in controllers and/or orchestrators, e.g., leveraging the rapidly growing eco-system for YANG tooling.

3.1.3. Extensibility

Path computation is only a subset of the typical functionality of a controller. In many use cases, issuing path computation requests comes along with the need to access other functionality on the same
system. In addition to obtaining TE topology, for instance also configuration of services (setup/modification/deletion) may be required, as well as:

1. Receiving notifications for topology changes as well as integration with fault management

2. Performance management such as retrieving monitoring and telemetry data

3. Service assurance, e.g., by triggering OAM functionality

4. Other fulfilment and provisioning actions beyond tunnels and services, such as changing QoS configurations

YANG is a very extensible and flexible data modeling language that can be used for all these use cases.

3.2. Interactions with TE Topology

The use cases described in section 2 have been described assuming that the topology view exported by each underlying SDN controller to the orchestrator is aggregated using the "virtual node model", defined in [RFC7926].

TE Topology information, e.g., as provided by [TE-TOPO], could in theory be used by an underlying SDN controllers to provide TE information to its client thus allowing a PCE available within its client to perform multi-domain path computation by its own, without requesting path computations to the underlying SDN controllers.

In case the client does not implement a PCE function, as discussed in section 1, it could not perform path computation based on TE Topology information and would instead need to request path computation to the underlying controllers to get the information it needs to compute the optimal end-to-end path.

This section analyzes the need for a client to request underlying SDN controllers for path computation even in case it implements a PCE functionality, as well as how the TE Topology information and the path computation can be complementary.

In nutshell, there is a scalability trade-off between providing all the TE information needed by PCE, when implemented by the client, to take optimal path computation decisions by its own versus sending
too many requests to underlying SDN Domain Controllers to compute a set of feasible optimal intra-domain TE paths.

3.2.1. TE Topology Aggregation

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export the whole TE domain as a single abstract TE node with a "detailed connectivity matrix".

The concept of a "detailed connectivity matrix" is defined in [TE-TOPO] to provide specific TE attributes (e.g., delay, SRLGs and summary TE metrics) as an extension of the "basic connectivity matrix", which is based on the "connectivity matrix" defined in [RFC7446].

The information provided by the "detailed connectivity matrix" would be equivalent to the information that should be provided by "virtual link model" as defined in [RFC7926].

For example, in the Packet/Optical integration use case, described in section 2.1, the Optical network controller can make the information shown in Figure 3 available to the Coordinator as part of the TE Topology information and the Coordinator could use this information to calculate by its own the optimal path between R1 and R2, without requesting any additional information to the Optical network Controller.

However, when designing the amount of information to provide within the "detailed connectivity matrix", there is a tradeoff to be considered between accuracy (i.e., providing "all" the information that might be needed by the PCE available to Orchestrator) and scalability.

Figure 6 below shows another example, similar to Figure 3, where there are two possible Optical paths between VP1 and VP4 with different properties (e.g., available bandwidth and cost).
Figure 6 - Packet/Optical Path Computation Example with multiple choices

Reporting all the information, as in Figure 6, using the "detailed connectivity matrix", is quite challenging from a scalability perspective. The amount of this information is not just based on number of end points (which would scale as N-square), but also on many other parameters, including client rate, user constraints/policies for the service, e.g. max latency < N ms, max cost, etc., exclusion policies to route around busy links, min OSNR margin, max preFEC BER etc. All these constraints could be different based on connectivity requirements.

Examples of how the "detailed connectivity matrix" can be dimensioned are described in Appendix A.

It is also worth noting that the "connectivity matrix" has been originally defined in WSON, [RFC7446], to report the connectivity constrains of a physical node within the WDM network: the information it contains is pretty "static" and therefore, once taken and stored in the TE data base, it can be always being considered valid and up-to-date in path computation request.

Using the "basic connectivity matrix" with an abstract node to abstract the information regarding the connectivity constraints of an Optical domain, would make this information more "dynamic" since the connectivity constraints of an Optical domain can change over
time because some optical paths that are feasible at a given time may become unfeasible at a later time when e.g., another optical path is established. The information in the "detailed connectivity matrix" is even more dynamic since the establishment of another optical path may change some of the parameters (e.g., delay or available bandwidth) in the "detailed connectivity matrix" while not changing the feasibility of the path.

The "connectivity matrix" is sometimes confused with optical reach table that contain multiple (e.g. k-shortest) regen-free reachable paths for every A-Z node combination in the network. Optical reach tables can be calculated offline, utilizing vendor optical design and planning tools, and periodically uploaded to the Controller: these optical path reach tables are fairly static. However, to get the connectivity matrix, between any two sites, either a regen free path can be used, if one is available, or multiple regen free paths are concatenated to get from src to dest, which can be a very large combination. Additionally, when the optical path within optical domain needs to be computed, it can result in different paths based on input objective, constraints, and network conditions. In summary, even though "optical reachability table" is fairly static, which regen free paths to build the connectivity matrix between any source and destination is very dynamic, and is done using very sophisticated routing algorithms.

There is therefore the need to keep the information in the "detailed connectivity matrix" updated which means that there another tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the client’s PCE) and having up-to-date information. The more the information is provided and the longer it takes to keep it up-to-date which increases the likelihood that the client’s PCE computes paths using not updated information.

It seems therefore quite challenging to have a "detailed connectivity matrix" that provides accurate, scalable and updated information to allow the client’s PCE to take optimal decisions by its own.

Instead, if the information in the "detailed connectivity matrix" is not complete/accurate, we can have the following drawbacks considering for example the case in Figure 6:
If only the VP1-VP4 path with available bandwidth of 2 Gb/s and cost 50 is reported, the client’s PCE will fail to compute a 5 Gb/s path between routers R1 and R2, although this would be feasible;

If only the VP1-VP4 path with available bandwidth of 10 Gb/s and cost 60 is reported, the client’s PCE will compute, as optimal, the 1 Gb/s path between R1 and R2 going through the VP2-VP5 path within the Optical domain while the optimal path would actually be the one going thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

Using the approach proposed in this document, the client, when it needs to setup an end-to-end path, it can request the Optical domain controller to compute a set of optimal paths (e.g., for VP1-VP4 and VP2-VP5) and take decisions based on the information received:

- When setting up a 5 Gb/s path between routers R1 and R2, the Optical domain controller may report only the VP1-VP4 path as the only feasible path; the Orchestrator can successfully setup the end-to-end path passing though this Optical path;

- When setting up a 1 Gb/s path between routers R1 and R2, the Optical domain controller (knowing that the path requires only 1 Gb/s) can report both the VP1-VP4 path, with cost 50, and the VP2-VP5 path, with cost 65. The Orchestrator can then compute the optimal path which is passing thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

### 3.2.2. TE Topology Abstraction

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export an abstract TE Topology, composed by a set of TE nodes and TE links, representing the abstract view of the topology controlled by each domain controller.

Considering the example in Figure 4, the TE domain controller 1 can export a TE Topology encompassing the TE nodes A, B, C and D and the TE Link interconnecting them. In a similar way, TE domain controller 2 can export a TE Topology encompassing the TE nodes E, F, G and H and the TE Link interconnecting them.

In this example, for simplicity reasons, each abstract TE node maps with each physical node, but this is not necessary.
In order to setup a multi-domain TE path (e.g., between nodes A and H), the multi-domain controller can compute by its own an optimal end-to-end path based on the abstract TE topology information provided by the domain controllers. For example:

- Multi-domain controller’s PCE, based on its own information, can compute the optimal multi-domain path being A-B-C-E-G-H, and then request the TE domain controllers to setup the A-B-C and E-G-H intra-domain paths.

- But, during path setup, the domain controller may find out that A-B-C intra-domain path is not feasible (as discussed in section 2.2, in optical networks it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller), while only the path A-B-D is feasible.

- So what the multi-domain controller computed is not good and need to re-start the path computation from scratch.

As discussed in section 3.2.1, providing more extensive abstract information from the TE domain controllers to the multi-domain controller may lead to scalability problems.

In a sense this is similar to the problem of routing and wavelength assignment within an Optical domain. It is possible to do first routing (step 1) and then wavelength assignment (step 2), but the chances of ending up with a good path is low. Alternatively, it is possible to do combined routing and wavelength assignment, which is known to be a more optimal and effective way for Optical path setup. Similarly, it is possible to first compute an abstract end-to-end path within the multi-domain Orchestrator (step 1) and then compute an intra-domain path within each Optical domain (step 2), but there are more chances not to find a path or to get a suboptimal path that performing per-domain path computation and then stitch them.

3.2.3. Complementary use of TE topology and path computation

As discussed in section 2.2, there are some scalability issues with path computation requests in a multi-domain TE network with many TE domains, in terms of the number of requests to send to the TE domain controllers. It would therefore be worthwhile using the TE topology information provided by the domain controllers to limit the number of requests.
An example can be described considering the multi-domain abstract topology shown in Figure 7. In this example, an end-to-end TE path between domains A and F needs to be setup. The transit domain should be selected between domains B, C, D and E.

![Figure 7 - Multi-domain with many domains (Topology information)](image-url)

The actual cost of each intra-domain path is not known a priori from the abstract topology information. The Multi-domain controller only knows, from the TE topology provided by the underlying domain controllers, the feasibility of some intra-domain paths and some upper-bound and/or lower-bound cost information. With this information, together with the cost of inter-domain links, the Multi-domain controller can understand by its own that:

- Domain B cannot be selected as the path connecting domains A and E is not feasible;

- Domain E cannot be selected as the path connecting domains A and E is not feasible.
Domain E cannot be selected as a transit domain since it is known from the abstract topology information provided by domain controllers that the cost of the multi-domain path A-E-F (which is 100, in the best case) will be always be higher than the cost of the multi-domain paths A-D-F (which is 90, in the worst case) and A-E-F (which is 80, in the worst case).

Therefore, the Multi-domain controller can understand by itself that the optimal multi-domain path could be either A-D-F or A-E-F but it cannot know which one of the two possible options actually provides the optimal end-to-end path.

The Multi-domain controller can therefore request path computation only to the TE domain controllers A, D, E and F (and not to all the possible TE domain controllers).

```
Figure 8  - Multi-domain with many domains (Path Computation information)
```

Based on these requests, the Multi-domain controller can know the actual cost of each intra-domain paths which belongs to potential
optimal end-to-end paths, as shown in Figure 8, and then compute the
optimal end-to-end path (e.g., A-D-F, having total cost of 50,
instead of A-C-F having a total cost of 70).

3.3. Stateless and Stateful Path Computation

The TE Tunnel YANG model, defined in [TE-TUNNEL], can support the
need to request path computation.

It is possible to request path computation by configuring a
"compute-only" TE tunnel and retrieving the computed path(s) in the
LSP(s) Record-Route Object (RRO) list as described in section 3.3.1
of [TE-TUNNEL].

This is a stateful solution since the state of each created
"compute-only" TE tunnel needs to be maintained and updated, when
underlying network conditions change.

It is very useful to provide options for both stateless and stateful
path computation mechanisms. It is suggested to use stateless
mechanisms as much as possible and to rely on stateful path
computation when really needed.

Stateless RPC allows requesting path computation using a simple
atomic operation and it is the natural option/choice, especially
with stateless PCE. The stateless path computation solution assumes
that the underlying SDN controller (e.g., a PNC) will compute a path
twice during the process to setup an LSP: at time T1, when its
client (e.g., an MDSC) sends a path computation RPC request to it,
and later, at time T2, when the same client (MDSC) creates a
tunnel requesting the setup of the LSP. The underlying assumption
is that, if network conditions have not changed, the same path that
has been computed at time T1 is also computed at time T2 by the
underlying SDN controller (e.g. PNC) and therefore the path that is
setup at time T2 is exactly the same path that has been computed at
time T1.

Since the operation is stateless, there is no guarantee that the
returned path would still be available when path setup is requested:
this does not cause major issues in case the time between path
computation and path setup is short (especially if compared with the
time that would be needed to update the information of a very
detailed connectivity matrix).
In most of the cases, there is even no need to guarantee that the path that has been setup is the exactly same as the path that has been returned by path computation, especially if it has the same or even better metrics. Depending on the abstraction level applied by the server, the client may also not know the actual computed path. The most important requirement is that the required global objectives (e.g., multi-domain path metrics and constraints) are met. For this reason a path verification phase is necessary to verify that the actual path that has been setup meets the global objectives (for example in a multi-domain network, the resulting end-to-end path meets the required end-to-end metrics and constraints).

In most of the cases, even if the setup path is not exactly the same as the path returned by path computation, its metrics and constraints are "good enough" (the path verification passes successfully). In the few corner cases where the path verification fails, it is possible repeat the whole process (path computation, path setup and path verification).

In case the stateless solution is not sufficient and it would be the need to setup at T2 exactly the same path computed at T1 a stateful solution, based on "compute-only" TE tunnel, could be used to get notifications in case the computed path has been changed. In this case at time T1, the client (MDSC) creates a te-tunnel in a compute-only mode in the config DS and later, at time T2, changes the configuration of that te-tunnel (not to be any more in a compute-only mode) to trigger the setup of the LSP.

It is worth noting that also the stateful solution, although increasing the likelihood that the computed path is available at path setup, does not guaranteed that because notifications may not be reliable or delivered on time. Path verification is needed also when stateful path computation is used.

The stateful path computation has also the following drawbacks:

- Several messages required for any path computation
- Requires persistent storage in the provider controller
- Need for garbage collection for stranded paths
3.3.1. Temporary reporting of the computed path state

This section describes an optional extension to the stateless behavior where the underlying SDN controller, after having received a path computation RPC request, maintains some "temporary state" associated with the computed path, allowing the client to request the setup of exactly that path, if still available.

This is similar to the stateful solution but, to avoid the drawbacks of the stateful approach, is leveraging the path computation RPC and the separation between configuration and operational DS, as defined in the NMDA architecture [RFC8342].

The underlying SDN controller, after having computed a path, as requested by a path computation RPC, also creates a te-tunnel instance within the operational DS, to store that computed path. This would be similar to the stateful solution with the only difference that there is no associated te-tunnel instance within the running DS.

Since underlying SDN controller stores in the operational DS the computed path based on an abstract topology it exposes, it also remembers, internally, which is the actual native path (physical path), within its native topology (physical topology), associated with that compute-only te-tunnel instance.

Afterwards, the client (e.g., MDSC) can request to setup that specific path by creating a te-tunnel instance (not in compute-only mode) in the running DS using the same tunnel-name of the existing te-tunnel in the operational datastore: this will trigger the underlying SDN controller to setup that path, if still available.

There are still cases where the path being setup is not exactly the same as the path that has been computed:

- When the tunnel is configured with path constraints which are not compatible with the computed path
- When the tunnel setup is requested after the resources of the computed path are no longer available
When the tunnel setup is requested after the computed path is no longer known (e.g. due to a server reboot) by the underlying SDN controller.

In all these cases, the underlying SDN controller should compute and setup a new path.

Therefore the "path verification" phase, as described in section 3.3 above, is still needed to check that the path that has been setup is still "good enough".

Since this new approach is not completely stateless, garbage collection is implemented using a timeout that, when it expires, triggers the removal of the computed path from the operational DS. This operation is fully controlled by the underlying SDN controller without the need for any action to be taken by the client that is not able to act on the operational datastore. The default value of this timeout is 10 minutes but a different value may be configured by the client.

In addition, it is possible for the client to tag each path computation requests with a transaction-id allowing for a faster removal of all the paths associated with a transaction-id, without waiting for their timers to expire.

The underlying SDN controller can remove from the operational DS all the paths computed with a given transaction-id which have not been setup either when it receives a Path Delete RPC request for that transaction-id or, automatically, right after the setup up of a path that have been previously computed with that transaction-id.

This possibility is useful when multiple paths are computed but, at most, only one is setup (e.g., in multi-domain path computation scenario scenarios). After the selected path has been setup (e.g, in one domain during multi-domain path setup), all the other alternative computed paths can be automatically deleted by the underlying SDN controller (since no longer needed). The client can also request, using the Path Delete RPC request, the underlying SDN controller to remove all the computed paths, if none of them is going to be setup (e.g., in a transit domain not being selected by multi-domain path computation and so not being automatically deleted).

This approach is complimentary and not alternative to the timer which is always needed to avoid stranded computed paths being stored.
in the operational DS when no path is setup and no explicit delete RPC is received.

4. Path Computation and Optimization for multiple paths

There are use cases, where it is advantageous to request path computation for a set of paths, through a network or through a network domain, using a single request [RFC5440].

In this case, sending a single request for multiple path computations, instead of sending multiple requests for each path computation, would reduce the protocol overhead and it would consume less resources (e.g., threads in the client and server).

In the context of a typical multi-domain TE network, there could multiple choices for the ingress/egress points of a domain and the Multi-domain controller needs to request path computation between all the ingress/egress pairs to select the best pair. For example, in the example of section 2.2, the Multi-domain controller needs to request the TE network controller 1 to compute the A-C and the A-D paths and to the TE network controller 2 to compute the E-H and the F-H paths.

It is also possible that the Multi-domain controller receives a request to setup a group of multiple end to end connections. The multi-domain controller needs to request each TE domain controller to compute multiple paths, one (or more) for each end to end connection.

There are also scenarios where it can be needed to request path computation for a set of paths in a synchronized fashion.

One example could be computing multiple diverse paths. Computing a set of diverse paths in a not-synchronized fashion, leads to the possibility of not being able to satisfy the diversity requirement. In this case, it is preferable to compute a sub-optimal primary path for which a diversely routed secondary path exists.

There are also scenarios where it is needed to request optimizing a set of paths using objective functions that apply to the whole set of paths, see [RFC5541], e.g. to minimize the sum of the costs of all the computed paths in the set.
5. YANG Model for requesting Path Computation

This document defines a YANG stateless RPC to request path computation as an "augmentation" of tunnel-rpc, defined in [TE-TUNNEL]. This model provides the RPC input attributes that are needed to request path computation and the RPC output attributes that are needed to report the computed paths.

```
augment /te:tunnels-rpc/te:input/te:tunnel-info:
    +---- path-request* [request-id] uint32
         ...........
      +--ro response* [response-id]
        +--ro response-id         uint32
        +--ro (response-type)?
          +--:(no-path-case)
            |     +--ro no-path!
          +--:(path-case)
            +--ro computed-path
                ...........
```

This model extensively re-uses the grouping defined in [TE-TUNNEL] to ensure maximal syntax and semantics commonality.

5.1. Synchronization of multiple path computation requests

The YANG model permits to synchronize a set of multiple path requests (identified by specific request-id) all related to a "svec" container emulating the syntax of "SVEC" PCEP object [RFC5440].

```
+---- synchronization* [synchronization-id]
    +---- synchronization-id     uint32
    +---- svec
    |    +---- relaxable?         boolean
    |    +---- disjointness?      te-path-disjointness
    |    +---- request-id-number* uint32
    +---- svec-constraints
    |    +---- path-metric-bound* [metric-type]
    |          +---- metric-type   identityref
    |          +---- upper-bound?  uint64
```

Busi, Belotti, et al. Expires September 11, 2019
The model, in addition to the metric types, defined in [TE-TUNNEL], which can be applied to each individual path request, defines additional specific metrics types that apply to a set of synchronized requests, as referenced in [RFC5541].

identity svec-metric-type {
  description
    "Base identity for svec metric type";
}

identity svec-metric-cumul-te {
  base svec-metric-type;
  description
    "TE cumulative path metric";
}

identity svec-metric-cumul-igp {
  base svec-metric-type;
}
description
  "IGP cumulative path metric";
}

identity svec-metric-cumul-hop {
  base svec-metric-type;
  description
    "Hop cumulative path metric";
}

identity svec-metric-aggregate-bandwidth-consumption {
  base svec-metric-type;
  description
    "Cumulative bandwith consumption of the set of synchronized paths";
}

identity svec-metric-load-of-the-most-loaded-link {
  base svec-metric-type;
  description
    "Load of the most loaded link";
}

5.2. Returned metric values

This YANG model provides a way to return the values of the metrics computed by the path computation in the output of RPC, together with other important information (e.g. srlg, affinities, explicit route), emulating the syntax of the "C" flag of the "METRIC" PCEP object [RFC5440]:

augment /te:tunnels-rpc/te:output/te:result:
  +--ro response* [response-id]
    +--ro response-id      uint32
    +--ro (response-type)?
      +--:(no-path-case)
      |    +--ro no-path!
      +--:(path-case)
        +--ro computed-path
        +--ro path-properties
It also allows to request in the input of RPC which information (metrics, srlg and/or affinities) should be returned:

```yang
augment /te:tunnels-rpc/te:input/te:tunnel-info:
  +---- path-request* [request-id]
      |      request-id                        uint32
      |      requested-metrics* [metric-type]
      |      return-srlgs?                    boolean
      |      return-affinities?               boolean
```

This feature is essential for using a stateless path computation in a multi-domain TE network as described in section 2.2. In this case, the metrics returned by a path computation requested to a given TE network controller must be used by the client to compute the best end-to-end path. If they are missing the client cannot compare...
different paths calculated by the TE network controllers and choose the best one for the optimal e2e path.

6. YANG model for stateless TE path computation

6.1. YANG Tree

Figure 9 below shows the tree diagram of the YANG model defined in module ietf-te-path-computation.yang.

```plaintext
module: ietf-te-path-computation
augment /te:tunnels-rpc/te:input/te:tunnel-info:
    +---- path-request* [request-id]
        | +---- request-id          uint32
        | +---- encoding?           identityref
        | +---- switching-type?     identityref
        | +---- source?             inet:ip-address
        | +---- destination?        inet:ip-address
        | +---- src-tp-id?          binary
        | +---- dst-tp-id?          binary
        | +---- bidirectional?      boolean
        +---- te-topology-identifier
            | +---- provider-id? te-global-id
            | +---- client-id? te-global-id
            | +---- topology-id? te-topology-id
            +---- explicit-route-objects-always
                | +---- route-object-exclude-always* [index]
                    |     +---- index          uint32
                    +---- (type)?
                        | +----:(numbered-node-hop)
                            |     +---- numbered-node-hop
                                |         +---- node-id te-node-id
                                +---- hop-type? te-hop-type
                        | +----:(numbered-link-hop)
                            +---- numbered-link-hop
                                +---- link-tp-id te-tp-id
                                +---- hop-type? te-hop-type
                                +---- direction? te-link-direction
                        | +----:(unnumbered-link-hop)
                            +---- unnumbered-link-hop
```
Internet-Draft  Yang for Path Computation  March 2019

+---- link-tp-id   te-tp-id
+---- node-id      te-node-id
+---- hop-type?    te-hop-type
+---- direction?   te-link-direction

++: as-number
    +---- as-number-hop
        +---- as-number    inet:as-number
            +---- hop-type?    te-hop-type

++: label
    +---- label-hop
        +---- te-label
            +---- (technology)?
            |    ++: (generic)
            |        +---- generic?
                rt-types:generalized-label
            +---- direction?   te-label-direction

++: route-object-include-exclude* [index]
    +---- explicit-route-usage?   identityref
    +---- index                   uint32
    +---- (type)?
        ++: numbered-node-hop
            +---- numbered-node-hop
                +---- node-id     te-node-id
                +---- hop-type?   te-hop-type
        ++: numbered-link-hop
            +---- numbered-link-hop
                +---- link-tp-id   te-tp-id
                +---- hop-type?   te-hop-type
                +---- direction?   te-link-direction
        ++: unnumbered-link-hop
            +---- unnumbered-link-hop
                +---- link-tp-id   te-tp-id
                +---- node-id      te-node-id
                +---- hop-type?    te-hop-type
                +---- direction?   te-link-direction
        ++: as-number-hop
            +---- as-number    inet:as-number
            +---- hop-type?    te-hop-type
Internet-Draft        Yang for Path Computation              March 2019

```yaml
+--:(label)
   |  +---- label-hop
   |     +---- te-label
   |        +---- (technology)?
   |            +--:(generic)
   |               +---- generic?
   |               |  +---- rt-types:generalized-label
   |               |     +---- direction?  te-label-direction
   |               +--:(srlg)
   |                 +---- srlg
   |                    +---- srlg?  uint32
+---- path-constraints
   +---- te-bandwidth
      |  +---- (technology)?
      |     +--:(generic)
      |        +---- generic?  te-bandwidth
      +---- link-protection?  identityref
      +---- setup-priority?  uint8
      +---- hold-priority?  uint8
      +---- signaling-type?  identityref
      +---- path-metric-bounds
         |  +---- path-metric-bound* [metric-type]
         |     |  +---- metric-type  identityref
         |     |  +---- upper-bound?  uint64
         +---- path-affinities-values
            |  +---- path-affinities-value* [usage]
            |     |  +---- usage  identityref
            |     +---- value?  admin-groups
            +---- path-affinity-names
               |  +---- path-affinity-name* [usage]
               |     |  +---- usage  identityref
               |     +---- affinity-name* [name]
               |          +---- name  string
               +---- path-srlgs-lists
                  |  +---- path-srlgs-list* [usage]
                  |     |  +---- usage  identityref
                  |     +---- values*  srlg
                  +---- path-srlgs-names
                     |  +---- path-srlgs-name* [usage]
```

Internet-Draft        Yang for Path Computation              March 2019

| | | +++ usage identityref
| | | +++ names* string
| | | +++ disjointness? te-path-disjointness
| | | +++ optimizations
| | | +++ (algorithm)?
| | | +--:(metric) (path-optimization-metric)?
| | | | +++ optimization-metric* [metric-type]
| | | | +++ metric-type
| | | identityref
| | | | +++ weight? uint8
| | | | +++ explicit-route-exclude-objects
| | | | | +++ route-object-exclude-object* [index]
| | | | | | +++ index uint32
| | | | | | +++ (type)?
| | | | | | +--:(numbered-node-hop)
| | | | | | | +++ numbered-node-hop
| | | | | | | | +++ node-id te-node-id
| | | | | | | | +++ hop-type? te-hop-type
| | | | | | +--:(numbered-link-hop)
| | | | | | | +++ numbered-link-hop
| | | | | | | | +++ link-tp-id te-tp-id
| | | | | | | | +++ hop-type? te-hop-type
| | | | | | | +++ direction? te-link-
| | | | | +--:(as-number)
| | | | | | +++ as-number-hop
| | | | | | | | +++ as-number inet:as-number
| | | | | | | | +++ hop-type? te-hop-type
| | | | | +--:(label)
| | | | | | +++ label-hop
| | | | | | | | +++ te-label
| | | | | | | | +++ (technology)?

| +---:(generic) |
| +----- generic? |
| rt- |
| types:generalized-label |
| +----- direction? |
| te-label-direction |
| +---:(srlg) |
| +----- srlg |
| +----- srlg? uint32 |
| +----- explicit-route-include-objects |
| +----- route-object-include-object* [index] |
| +----- index uint32 |
| +----- (type)? |
| +---:(numbered-node-hop) |
| +----- numbered-node-hop |
| +----- node-id te-node-id |
| +----- hop-type? te-hop-type |
| +---:(numbered-link-hop) |
| +----- numbered-link-hop |
| +----- link-tp-id te-tp-id |
| +----- hop-type? te-hop-type |
| +----- direction? te-link- |
| direction |
| +---:(unnumbered-link-hop) |
| +----- unnumbered-link-hop |
| +----- link-tp-id te-tp-id |
| +----- node-id te-node-id |
| +----- hop-type? te-hop-type |
| +----- direction? te-link- |
| direction |
| +---:(as-number) |
| +----- as-number-hop |
| +----- as-number inet:as-number |
| +----- hop-type? te-hop-type |
| +---:(label) |
| +----- label-hop |
| +----- te-label |
| +----- (technology)? |
| +---:(generic)|
| | | | | | generic? |
| | | | | direction? |
| | | | | tiebreakers |
| | | | | tiebreaker* [tiebreaker-type] |
| | | | | tiebreaker-type identityref |
| | | | | (objective-function) |
| | | | | (path-optimization-objective-function)? |
| | | | | objective-function |
| | | | | objective-function-type? identityref |
| | | | path-in-segment! |
| | | | label-restrictions |
| | | | label-restriction* [index] |
| | | | restriction? enumeration |
| | | | index uint32 |
| | | | label-start |
| | | | te-label |
| | | | (technology)? |
| | | | : (generic) |
| | | | generic? rt-types:generalized-label |
| | | | direction? te-label-direction |
| | | | label-end |
| | | | te-label |
| | | | (technology)? |
| | | | : (generic) |
| | | | generic? rt-types:generalized-label |
| | | | direction? te-label-direction |
| | | | label-step |
| | | | (technology)? |
| | | | : (generic) |
| | | | generic? int32 |
| | | | range-bitmap? yang:hex-string |
| | | path-out-segment! |
| | | label-restrictions |
| | | label-restriction* [index] |
| +---- request-id-number*       uint32
| +---- svec-constraints
|   +---- path-metric-bound* [metric-type]
|     +---- metric-type    identityref
|     +---- upper-bound?   uint64
| +---- path-srlgs-lists
|   +---- path-srlgs-list* [usage]
|     +---- usage    identityref
|     +---- values*   srlg
| +---- path-srlgs-names
|   +---- path-srlgs-name* [usage]
|     +---- usage    identityref
|     +---- names*   string
| +---- exclude-objects
|   +---- excludes* [index]
|     +---- index       uint32
|     +---- (type)?
|       +--:(numbered-node-hop)
|         +---- numbered-node-hop
|           +---- node-id     te-node-id
|           +---- hop-type?   te-hop-type
|       +--:(numbered-link-hop)
|         +---- numbered-link-hop
|           +---- link-tp-id    te-tp-id
|           +---- hop-type?   te-hop-type
|           +---- direction?  te-link-direction
|       +--:(unnumbered-link-hop)
|         +---- unnumbered-link-hop
|           +---- link-tp-id    te-tp-id
|           +---- node-id      te-node-id
|           +---- hop-type?    te-hop-type
|           +---- direction?   te-link-direction
|       +--:(as-number)
|         +---- as-number-hop
|           +---- as-number    inet:as-number
|           +---- hop-type?    te-hop-type
|       +--:(label)
|         +---- label-hop
|           +---- te-label
Internet-Draft Yang for Path Computation March 2019

---ro usage identityref
---ro values* srlg
---ro path-srlgs-names
  ---ro path-srlgs-name* [usage]
    ---ro usage identityref
    ---ro names* string
---ro path-route-objects
  ---ro path-route-object* [index]
    ---ro index uint32
    ---ro (type)?
      ---:(numbered-node-hop)
      |   ---ro numbered-node-hop
      |     ---ro node-id te-node-id
      |     ---ro hop-type? te-hop-type
      ---:(numbered-link-hop)
      |   ---ro numbered-link-hop
      |     ---ro link-tp-id te-tp-id
      |     ---ro hop-type? te-hop-type
      |     ---ro direction? te-link-
      direction
      ---:(unnumbered-link-hop)
      |   ---ro unnumbered-link-hop
      |     ---ro link-tp-id te-tp-id
      |     ---ro node-id te-node-id
      |     ---ro hop-type? te-hop-type
      |     ---ro direction? te-link-
      direction
      ---:(as-number)
      |   ---ro as-number-hop
      |     ---ro as-number inet:as-number
      |     ---ro hop-type? te-hop-type
      ---:(label)
      |   ---ro label-hop
      |     ---ro te-label
      |     ---ro (technology)?
      |       ---:(generic)
      |         ---ro generic? rt-

types:generalized-label

Internet-Draft        Yang for Path Computation              March 2019

Figure 9 - TE path computation YANG tree

6.2. YANG Module

<CODE BEGINS>file "ietf-te-path-computation@2019-03-11.yang"
module ietf-te-path-computation {
  yang-version 1.1;
  // replace with IANA namespace when assigned
  prefix "tepc";

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

  import ietf-te {
    prefix "te";
  }

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

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

Busi, Belotti, et al. Expires September 11, 2019
contact
  "WG Web:  <http://tools.ietf.org/wg/teas/>
  WG List:  <mailto:teas@ietf.org>

  WG Chair: Lou Berger
  <mailto:lberger@labn.net>

  WG Chair: Vishnu Pavan Beeram
  <mailto:vbeeram@juniper.net>

",

description "YANG model for stateless TE path computation";

revision "2019-03-11" {
  description
    "Initial revision";
  reference
    "draft-ietf-teas-yang-path-computation";
}

/* Features */

feature stateless-path-computation {
  description
    "This feature indicates that the system supports
     stateless path computation.";
}

/* Groupings */

grouping path-info {
  uses te-types:generic-path-properties;
  description "Path computation output information";
}
grouping requested-info {
    description
    "This grouping defines the information (e.g., metrics) which must be returned in the response";
    list requested-metrics {
        key 'metric-type';
        description
        "The list of the requested metrics. The metrics listed here must be returned in the response. Returning other metrics in the response is optional.";
        leaf metric-type {
            type identityref {
                base te-types:path-metric-type;
            }
            description
            "The metric that must be returned in the response";
        }
    }
    leaf return-srlgs {
        type boolean;
        default false;
        description
        "If true, path srlgs must be returned in the response. If false, returning path srlgs in the response is optional.";
    }
    leaf return-affinities {
        type boolean;
        default false;
        description
        "If true, path affinities must be returned in the response. If false, returning path affinities in the response is optional.";
    }
}

grouping requested-state {
    description
"Configuration for the transient state used to report the computed path";
leaf timer {
    type uint16;
    units minutes;
    default 10;
    description "The timeout after which the transient state reporting the computed path should be removed.";
}
leaf transaction-id {
    type string;
    description "The transaction-id associated with this path computation to be used for fast deletion of the transient states associated with multiple path computations. This transaction-id can be used to explicitly delete all the transient states of all the computed paths associated with the same transaction-id.

When one path associated with a transaction-id is setup, the transient states of all the other computed paths with the same transaction-id are automatically removed.

If not specified, the transient state is removed only when the timer expires (when the timer is specified) or not created at all (stateless path computation, when the timer is not specified).
"
}
leaf tunnel-name {
    type string;
    description "The suggested name to be assigned to the te-tunnel instance which is created to report the computed path."
In case multiple paths are requested with the same suggested name, the server will create only one te-tunnel instance to report all the computed paths with the same suggested name.

A different name can be assigned by server (e.g., when a te-tunnel with this name already exists).

```yaml
choice path {
  description "The transient state of the computed path can be reported as a primary or a secondary path of a te-tunnel";
  case primary {
    leaf primary-path-name {
      type string;
      description "The suggested name to be assigned to the p2p-primary-path instance which is created to report the computed path.

      A different name can be assigned by the server (e.g., when a p2p-primary-path with this name already exists)."
    }
  }
  case secondary {
    leaf secondary-path-name {
      type string;
      description "The suggested name to be assigned to the p2p-secondary-path instance which is created to report the computed path.

      A different name can be assigned by the server (e.g., when a p2p-secondary-path with this
```
name already exists).

If not specified, the a p2p-primary-path is created by the server.
";}
}
}
}
grouping reported-state {
    description
    "Information about the transient state created to report the computed path";

    leaf tunnel-ref {
        type te:tunnel-ref;
        description
        "Reference to the tunnel that reports the transient state of the computed path.
        If no transient state is created, this attribute is empty.";
    }

    choice path {
        description
        "The transient state of the computed path can be reported as a primary or a secondary path of a te-tunnel";

        case primary {
            leaf primary-path-ref {
                type leafref {
                    path "/te:te/te:tunnels/" +
                      "te:tunnel[te:name=current()]/../tunnel-ref]/" +
                      "te:p2p-primary-paths/te:p2p-primary-path/" +
                      "te:name";
                }
                must ".../tunnel-ref" {
                    description
"The primary-path-name can only be reported if also the tunnel is reported to provide the complete reference.";

} description
"Reference to the p2p-primary-path that reports the transient state of the computed path.

If no transient state is created, this attribute is empty."
}
}

} case secondary {
leaf secondary-path-ref {
  type leafref {
    path "/te:te/te:tunnels/" +
    "te:tunnel[te:name=current()../tunnel-ref]/" +
    "te:p2p-secondary-paths/te:p2p-secondary-path/" +
    "te:name";
  }
  must ".../tunnel-ref" {
    description
    "The secondary-path-name can only be reported if also the tunnel is reported to provide the complete reference.";
  }
  description
  "Reference to the p2p-secondary-path that reports the transient state of the computed path.

If no transient state is created, this attribute is empty."
  }
  }
}
identity svec-metric-type {
    description
    "Base identity for svec metric type";
}

identity svec-metric-cumul-te {
    base svec-metric-type;
    description
    "TE cumulative path metric";
}

identity svec-metric-cumul-igp {
    base svec-metric-type;
    description
    "IGP cumulative path metric";
}

identity svec-metric-cumul-hop {
    base svec-metric-type;
    description
    "Hop cumulative path metric";
}

identity svec-metric-aggregate-bandwidth-consumption {
    base svec-metric-type;
    description
    "Cumulative bandwith consumption of the set of synchronized paths";
}

identity svec-metric-load-of-the-most-loaded-link {
    base svec-metric-type;
    description
    "Load of the most loaded link";
}

grouping svec-metrics-bounds_config {

"TE path metric bounds grouping for computing a set of synchronized requests";
leaf metric-type {
    type identityref {
        base svec-metric-type;
    }
    description "TE path metric type usable for computing a set of synchronized requests";
}
leaf upper-bound {
    type uint64;
    description "Upper bound on end-to-end svec path metric";
}

"TE path metric bounds grouping for computing a set of synchronized requests";
leaf metric-type {
    type identityref {
        base svec-metric-type;
    }
    description "TE path metric type usable for computing a set of synchronized requests";
}
leaf weight {
    type uint8;
    description "Metric normalization weight";
}

"List of resources to be excluded by all the paths in the SVEC";
container exclude-objects {
    description "resources to be excluded";
list excludes {
  key index;
  ordered-by user;
  leaf index {
    type uint32;
    description "XRO subobject index";
  }
  description
  "List of explicit route objects to always exclude
  from synchronized path computation";
  uses te-types:explicit-route-hop;
}

grouping synchronization-constraints {
  description "Global constraints applicable to synchronized
  path computation";
  container svec-constraints {
    description "global svec constraints";
    list path-metric-bound {
      key metric-type;
      description "list of bound metrics";
      uses svec-metrics-bounds_config;
    }
  }
  uses te-types:generic-path-srlgs;
  uses svec-exclude;
}

grouping synchronization-optimization {
  description "Synchronized request optimization";
  container optimizations {
    description
    "The objective function container that includes attributes
    to impose when computing a synchronized set of paths";
    choice algorithm {
      description "Optimizations algorithm.";
    }
}
case metric {
    if-feature te-types:path-optimization-metric;
    list optimization-metric {
        key "metric-type";
        description "svec path metric type";
        uses svect-metrics-optimization_config;
    }
}


case objective-function {
    if-feature te-types:path-optimization-objective-function;
    container objective-function {
        description "The objective function container that includes attributes to impose when computing a TE path";
        leaf objective-function-type {
            type identityref {
                base te-types:objective-function-type;
            }
            default te-types:of-minimize-cost-path;
            description "Objective function entry";
        }
    }
}


grouping synchronization-info {
    description "Information for sync";
    list synchronization {
        key "synchronization-id";
        description "sync list";
        leaf synchronization-id {
            type uint32;
            description "index";
        }
        container svec {
            description "Synchronization VECTro";
        }
    }
}
leaf relaxable {
    type boolean;
    default true;
    description
    "If this leaf is true, path computation process is free to ignore svec content.
    Otherwise, it must take into account this svec."
}
uses te-types:generic-path-disjointness;
leaf-list request-id-number {
    type uint32;
    description
    "This list reports the set of path computation requests that must be synchronized.";
}
uses synchronization-constraints;
uses synchronization-optimization;
}

grouping no-path-info {
    description "no-path-info";
    container no-path {
        presence "Response without path information, due to failure performing the path computation";
        description "if path computation cannot identify a path, rpc returns no path.";
    }
}

/*
 * These groupings should be removed when defined in te-types
 */
grouping encoding-and-switching-type {
    description
    "Common grouping to define the LSP encoding and switching types";
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";
}

grouping tunnel-p2p-common-params {
  description "Common grouping to define the TE tunnel parameters";
  uses encoding-and-switching-type;
  leaf source {
    type inet:ip-address;
    description "TE tunnel source address.";
  }
  leaf destination {
    type inet:ip-address;
    description "P2P tunnel destination address";
  }
  leaf src-tp-id {
    type binary;
    description "TE tunnel source termination point identifier.";
  }
  leaf dst-tp-id {
    type binary;
    description "TE tunnel destination termination point identifier.";
  }
}
leaf bidirectional {
  type boolean;
  default 'false';
  description "TE tunnel bidirectional";
}

/*
* AUGMENTS TO TE RPC
*/

augment "/te:tunnels-rpc/te:input/te:tunnel-info" {
  description "Path Computation RPC input";
  list path-request {
    key "request-id";
    description "request-list";
    leaf request-id {
      type uint32;
      mandatory true;
      description 
        "Each path computation request is uniquely identified 
         by the request-id-number.";
    }
    uses tunnel-p2p-common-params;
    uses te-types:te-topology-identifier;
    uses te-types:path-constraints-route-objects;
    uses te-types:generic-path-constraints;
    uses te-types:generic-path-optimization;
    uses te:path-access-segment-info;
    uses requested-info;
    container requested-state {
      presence 
        "Request temporary reporting of the computed path state";
      description 
        "Configures attributes for the temporary reporting of the 
         computed path state (e.g., expiration timer).";
      uses requested-state;
    }
  }
}
augment "/te:tunnels-rpc/te:output/te:result" {
  description "Path Computation RPC output";
  list response {
    key "response-id";
    config false;
    description "response";
    leaf response-id {
      type uint32;
      description "The response-id has the same value of the corresponding request-id.";
    }
    choice response-type {
      config false;
      description "response-type";
      case no-path-case {
        uses no-path-info;
      }
      case path-case {
        container computed-path {
          uses path-info;
          uses reported-state;
          description "Path computation service.";
        }
      }
    }
  }
}

augment "/te:tunnels-rpc/te:input/te:tunnel-info" {
  description "Path Delete RPC input";
  leaf-list deleted-paths-transaction-id {
    type string;
    description "The list of the transaction-id values of the";
  }
}
transient states to be deleted";
}
}

augment "/te:tunnels-rpc/te:output/te:result" {
  description "Path Delete RPC output";
  leaf-list deleted-paths-transaction-id {
    type string;
    description
      "The list of the transaction-id values of the transient states that have been successfully deleted";
  }
}

<CODE ENDS>

Figure 10 - TE path computation YANG module

7. Security Considerations

This document describes use cases of requesting Path Computation using YANG models, which could be used at the ABNO Control Interface [RFC7491] and/or between controllers in ACTN [RFC8453]. As such, it does not introduce any new security considerations compared to the ones related to YANG specification, ABNO specification and ACTN Framework defined in [RFC7950], [RFC7491] and [RFC8453].

The YANG module defined in this draft is designed to be accessed via the NETCONF protocol [RFC6241] or RESTCONF protocol [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

This document also defines common data types using the YANG data modeling language. The definitions themselves have no security impact on the Internet, but the usage of these definitions in concrete YANG modules might have. The security considerations spelled out in the YANG specification [RFC7950] apply for this document as well.
The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

Note - The security analysis of each leaf is for further study.

8. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC7950].

name: ietf-te-path-computation
prefix: tepc

9. References

9.1. Normative References


9.1. Informative References


Busi, Belotti, et al. Expires September 11, 2019
10. Acknowledgments

The authors would like to thank Igor Bryskin and Xian Zhang for participating in the initial discussions that have triggered this work and providing valuable insights.

The authors would like to thank the authors of the TE Tunnel YANG model [TE-TUNNEL], in particular Igor Bryskin, Vishnu Pavan Beeram, Tarek Saad and Xufeng Liu, for their inputs to the discussions and support in having consistency between the Path Computation and TE Tunnel YANG models.

The authors would like to thank Adrian Farrel, Dhruv Dhody, Igor Bryskin, Julien Meuric and Lou Berger for their valuable input to the discussions, which has clarified that the path being setup is not necessarily the same as the path that have been previously computed and, in particular to Dhruv Dhody, for his suggestion to describe the need for a path verification phase to check that the actual path being setup meets the required end-to-end metrics and constraints.

This document was prepared using 2-Word-v2.0.template.dot.
Appendix A. Examples of dimensioning the "detailed connectivity matrix"

In the following table, a list of the possible constraints, associated with their potential cardinality, is reported.

The maximum number of potential connections to be computed and reported is, in first approximation, the multiplication of all of them.

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>End points</td>
<td>$N(N-1)/2$ if connections are bidirectional (OTN and WDM), $N(N-1)$ for unidirectional connections.</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>In WDM networks, bandwidth values are expressed in GHz.</td>
</tr>
</tbody>
</table>

On fixed-grid WDM networks, the central frequencies are on a 50GHz grid and the channel width of the transmitters are typically 50GHz such that each central frequency can be used, i.e., adjacent channels can be placed next to each other in terms of central frequencies.

On flex-grid WDM networks, the central frequencies are on a 6.25GHz grid and the channel width of the transmitters can be multiples of 12.5GHz.

For fixed-grid WDM networks typically there is only one possible bandwidth value (i.e., 50GHz) while for flex-grid WDM networks typically there are 4 possible bandwidth values (e.g., 37.5GHz, 50GHz, 62.5GHz, 75GHz).

In OTN (ODU) networks, bandwidth values are expressed as pairs of ODU type and, in case of ODUflex, ODU rate in bytes/sec as described in section 5 of [RFC7139].

For "fixed" ODUk types, 6 possible bandwidth values are possible (i.e., ODU0, ODU1, ODU2, ODU2e, ODU3, ODU4).

For ODUflex(GFP), up to 80 different bandwidth values can be specified, as defined in Table 7-8 of [ITU-T G.709-2016].

For other ODUflex types, like ODUflex(CBR), the number of possible bandwidth values depends on the rates of the
clients that could be mapped over these ODUflex types, as shown in Table 7.2 of [ITU-T G.709-2016], which in theory could be a continuum of values. However, since different ODUflex bandwidths that use the same number of TSs on each link along the path are equivalent for path computation purposes, up to 120 different bandwidth ranges can be specified.

Ideas to reduce the number of ODUflex bandwidth values in the detailed connectivity matrix, to less than 100, are for further study.

Bandwidth specification for ODUCn is currently for further study but it is expected that other bandwidth values can be specified as integer multiples of 100Gb/s.

In IP we have bandwidth values in bytes/sec. In principle, this is a continuum of values, but in practice we can identify a set of bandwidth ranges, where any bandwidth value inside the same range produces the same path. The number of such ranges is the cardinality, which depends on the topology, available bandwidth and status of the network. Simulations (Note: reference paper submitted for publication) show that values for medium size topologies (around 50-150 nodes) are in the range 4-7 (5 on average) for each end points couple.

Metrics
IGP, TE and hop number are the basic objective metrics defined so far. There are also the 2 objective functions defined in [RFC5541]: Minimum Load Path (MLP) and Maximum Residual Bandwidth Path (MBP). Assuming that one only metric or objective function can be optimized at once, the total cardinality here is 5.

With [RFC8233], a number of additional metrics are defined, including Path Delay metric, Path Delay Variation metric and Path Loss metric, both for point-to-point and point-to-multipoint paths. This increases the cardinality to 8.

Bounds
Each metric can be associated with a bound in order to find a path having a total value of that metric lower than the given bound. This has a potentially very high cardinality (as any value for the bound is allowed). In
practice there is a maximum value of the bound (the one with the maximum value of the associated metric) which results always in the same path, and a range approach like for bandwidth in IP should produce also in this case the cardinality. Assuming to have a cardinality similar to the one of the bandwidth (let say 5 on average) we should have 6 (IGP, TE, hop, path delay, path delay variation and path loss; we don’t consider here the two objective functions of [RFC5541] as they are conceived only for optimization)*5 = 30 cardinality.

Technology constraints For further study

Priority We have 8 values for setup priority, which is used in path computation to route a path using free resources and, where no free resources are available, resources used by LSPs having a lower holding priority.

Local prot It’s possible to ask for a local protected service, where all the links used by the path are protected with fast reroute (this is only for IP networks, but line protection schemas are available on the other technologies as well). This adds an alternative path computation, so the cardinality of this constraint is 2.

Administrative Colors Administrative colors (aka affinities) are typically assigned to links but when topology abstraction is used affinity information can also appear in the detailed connectivity matrix.

There are 32 bits available for the affinities. Links can be tagged with any combination of these bits, and path computation can be constrained to include or exclude any or all of them. The relevant cardinality is 3 (include-any, exclude-any, include-all) times $2^{32}$ possible values. However, the number of possible values used in real networks is quite small.

Included Resources

A path computation request can be associated to an ordered set of network resources (links, nodes) to be included along the computed path. This constraint would
have a huge cardinality as in principle any combination of network resources is possible. However, as far as the Orchestrator doesn’t know details about the internal topology of the domain, it shouldn’t include this type of constraint at all (see more details below).

Excluded Resources

A path computation request can be associated to a set of network resources (links, nodes, SRLGs) to be excluded from the computed path. Like for included resources, this constraint has a potentially very high cardinality, but, once again, it can’t be actually used by the Orchestrator, if it’s not aware of the domain topology (see more details below).

As discussed above, the Orchestrator can specify include or exclude resources depending on the abstract topology information that the domain controller exposes:

- In case the domain controller exposes the entire domain as a single abstract TE node with his own external terminations and detailed connectivity matrix (whose size we are estimating), no other topological details are available, therefore the size of the detailed connectivity matrix only depends on the combination of the constraints that the Orchestrator can use in a path computation request to the domain controller. These constraints cannot refer to any details of the internal topology of the domain, as those details are not known to the Orchestrator and so they do not impact size of the detailed connectivity matrix exported.
Instead in case the domain controller exposes a topology including more than one abstract TE nodes and TE links, and their attributes (e.g. SRLGs, affinities for the links), the Orchestrator knows these details and therefore could compute a path across the domain referring to them in the constraints. The detailed connectivity matrixes, whose size need to be estimated here, are the ones relevant to the abstract TE nodes exported to the Orchestrator. These detailed connectivity matrixes and therefore theirs sizes, while cannot depend on the other abstract TE nodes and TE links, which are external to the given abstract node, could depend to SRLGs (and other attributes, like affinities) which could be present also in the portion of the topology represented by the abstract nodes, and therefore contribute to the size of the related detailed connectivity matrix.

We also don’t consider here the possibility to ask for more than one path in diversity or for point-to-multi-point paths, which are for further study.

Considering for example an IP domain without considering SRLG and affinities, we have an estimated number of paths depending on these estimated cardinalities:

Endpoints = N*(N-1), Bandwidth = 5, Metrics = 6, Bounds = 20, Priority = 8, Local prot = 2

The number of paths to be pre-computed by each IP domain is therefore 24960 * N(N-1) where N is the number of domain access points.

This means that with just 4 access points we have nearly 300000 paths to compute, advertise and maintain (if a change happens in the domain, due to a fault, or just the deployment of new traffic, a substantial number of paths need to be recomputed and the relevant changes advertised to the upper controller).

This seems quite challenging. In fact, if we assume a mean length of 1K for the json describing a path (a quite conservative estimate), reporting 300000 paths means transferring and then parsing more than 300 Mbytes for each domain. If we assume that 20% (to be checked) of this paths change when a new deployment of traffic occurs, we have 60 Mbytes of transfer for each domain traversed by a new end-to-end path. If a network has, let say, 20 domains (we want to estimate the load for a non-trivial domain setup) in the beginning a total...
initial transfer of 6Gigs is needed, and eventually, assuming 4-5 domains are involved in mean during a path deployment we could have 240-300 Mbytes of changes advertised to the higher order controller.

Further bare-bone solutions can be investigated, removing some more options, if this is considered not acceptable; in conclusion, it seems that an approach based only on the information provided by the detailed connectivity matrix is hardly feasible, and could be applicable only to small networks with a limited meshing degree between domains and renouncing to a number of path computation features.

Contributors

Dieter Beller
Nokia
Email: dieter.beller@nokia.com

Gianmarco Bruno
Ericsson
Email: gianmarco.bruno@ericsson.com

Francesco Lazzeri
Ericsson
Email: francesco.lazzeri@ericsson.com

Young Lee
Huawei
Email: leeyoung@huawei.com

Carlo Perocchio
Ericsson
Email: carlo.perocchio@ericsson.com

Authors’ Addresses

Italo Busi (Editor)
Huawei
Email: italo.busi@huawei.com
A YANG Data Model for Resource Reservation Protocol (RSVP)
draft-ietf-teas-yang-rsvp-10

Abstract

This document defines a YANG data model for the configuration and management of RSVP Protocol. The model covers the building blocks of the RSVP protocol that can be augmented and used by other RSVP extension models such as RVSP extensions to Traffic-Engineering (RSVP-TE). The model covers the configuration, operational state, remote procedural calls, and event notifications data.

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 August 22, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g., ReST) and encoding other than XML (e.g., JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage the RSVP protocol [RFC2205]. This model covers RSVP protocol building blocks that can be augmented and used by other RSVP extension models—such as for signaling RSVP-TE MPLS (or other technology specific) Label Switched Paths (LSP)s.
1.1. Terminology

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

The terminology for describing YANG data models is found in [RFC7950].

1.2. Model Tree Diagram

A full tree diagram of the module(s) defined in this document is given in subsequent sections as per the syntax defined in [RFC8340].

1.3. Prefixes in Data Node Names

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>rt-type</td>
<td>ietf-routing-types</td>
<td>XX</td>
</tr>
<tr>
<td>key-chain</td>
<td>ietf-key-chain</td>
<td>XX</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

2. Model Overview

The RSVP base YANG module augments the "control-plane-protocol" list in ietf-routing [RFC8349] module with specific RSVP parameters in an "rsvp" container. It also defines an extension identity "rsvp" of base "rtr:routing-protocol" to identify the RSVP protocol.

The augmentation of the RSVP model by other models (e.g. RSVP-TE for MPLS or other technologies) are outside the scope of this document and are discussed in separate document(s), e.g. [I-D.ietf-teas-yang-rsvp-te].
2.1. Module(s) Relationship

This document divides the RSVP model into two modules: base and extended RSVP modules. Some RSVP features are categorized as core to the function of the protocol that are supported by most vendors claiming support for RSVP protocol. Such features configuration and state are grouped in the RSVP base module.

Other extended RSVP features are categorized as either optional or providing knobs to better tune basic functionality of the RSVP protocol. The support for extended RSVP features by all vendors is considered optional. Such features are grouped in a separate RSVP extended module.

The relationship between the base and extended RSVP YANG model and the IETF routing YANG model is shown in Figure 1.

```
+--------------+
Routing        | ietf-routing |
+--------------+
    +----------+
    |          |
    +----------+
RSVP module    | ietf-rsvp  |
    +----------+
          +------+
          |      |
          +------+
RSVP extended | ietf-rsvp-extended |
    +------------------+
                    +----------+
                    |          |
                    +----------+
```

Figure 1: Relationship of RSVP and RSVP extended modules with other protocol modules

2.2. Design Considerations

The RSVP base model does not aim to be feature complete. The primary intent is to cover a set of standard core features that are commonly in use. For example:

- Authentication ([RFC2747])
- Refresh Reduction ([RFC2961])
- Hellos ([RFC3209])
- Graceful Restart ([RFC3473], [RFC5063])
The extended RSVP YANG model covers non-basic configuration(s) for RSVP feature(s) as well as optional RSVP feature that are not a must for basic RSVP operation.

The defined data model supports configuration inheritance for neighbors, and interfaces. Data elements defined in the main container (e.g. the container that encompasses the list of interfaces, or neighbors) are assumed to apply equally to all elements of the list, unless overridden explicitly for a certain element (e.g. interface). Vendors are expected to augment the above container(s) to provide the list of inheritance command for their implementations.

2.3. RSVP Base YANG Model

The RSVP base YANG data model defines the container "rsvp" as the top level container in this data model. The presence of this container enables the RSVP protocol functionality.

Derived state data is contained under a "state" container of the intended object as shown in Figure 2.
The following subsections provide overview of the parts of the model pertaining to configuration and state data.

Configuration and state data are organized into those applicable globally (node scope), per interface, per neighbor, or per session.

Global Data:

The global data branch of the model covers configuration and state that are applicable the RSVP protocol behavior.

Interface Data:
The interface data branch of the data model covers configuration and state elements relevant to one or all RSVP interfaces. Any data configuration applied at the "interfaces" container level are equally applicable to all interfaces - unless overridden by explicit configuration under a specific interface.

Neighbor Data:

The neighbor data branch of the data model covers configuration and state elements relevant to RSVP neighbors.

Session Data:

The sessions data branch covers configuration of elements relevant to RSVP sessions.

2.3.1. Tree Diagram

Figure 3 shows the YANG tree representation for configuration and state data that is augmenting the RSVP basic module:

```
module: ietf-rsvp
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol:
    +--rw rsvp!
    +--rw globals
      +--rw sessions
        +--ro session* [local-index]
          +--ro local-index  -> ../state/local-index
          +--ro state
            +--ro local-index? uint64
            +--ro destination-port? inet:port-number
            +--ro source? inet:ip-address
            +--ro destination? inet:ip-address
            +--ro session-name? string
            +--ro session-state? enumeration
            +--ro session-type? identityref
            +--ro psbs
              +--ro psb* []
                +--ro source-port? inet:port-number
                +--ro expires-in? uint32
              +--ro rsbs
                +--ro rsb* []
                  +--ro source-port? inet:port-number
                  +--ro reservation-style? identityref
                  +--ro expires-in? uint32
          +--rw statistics
            +--ro state
```
Internet-Draft RSVP YANG Data Model February 2019

++-ro messages
    +++-ro ack-sent? yang:counter64
    +++-ro ack-received? yang:counter64
    +++-ro bundle-sent? yang:counter64
    +++-ro bundle-received? yang:counter64
    +++-ro hello-sent? yang:counter64
    +++-ro hello-received? yang:counter64
    +++-ro integrity-challenge-sent? yang:counter64
    +++-ro integrity-challenge-received? yang:counter64
    +++-ro integrity-response-sent? yang:counter64
    +++-ro integrity-response-received? yang:counter64
    +++-ro notify-sent? yang:counter64
    +++-ro notify-received? yang:counter64
    +++-ro path-sent? yang:counter64
    +++-ro path-received? yang:counter64
    +++-ro path-err-sent? yang:counter64
    +++-ro path-err-received? yang:counter64
    +++-ro path-tear-sent? yang:counter64
    +++-ro path-tear-received? yang:counter64
    +++-ro resv-sent? yang:counter64
    +++-ro resv-received? yang:counter64
    +++-ro resv-confirm-sent? yang:counter64
    +++-ro resv-confirm-received? yang:counter64
    +++-ro resv-err-sent? yang:counter64
    +++-ro resv-err-received? yang:counter64
    +++-ro resv-tear-sent? yang:counter64
    +++-ro resv-tear-received? yang:counter64
    +++-ro summary-refresh-sent? yang:counter64
    +++-ro summary-refresh-received? yang:counter64
++-ro unknown-messages-received? yang:counter64
++-ro packets
    +++-ro sent? yang:counter64
    +++-ro received? yang:counter64
++-ro errors
    +++-ro authenticate? yang:counter64
    +++-ro checksum? yang:counter64
    +++-ro packet-len? yang:counter64
++-rw graceful-restart
    +++-rw enabled? boolean
++-rw interfaces
    +++-rw refresh-reduction
        +++-rw enabled? boolean
    +++-rw hellos
        +++-rw enabled? boolean
    +++-rw authentication
        +++-rw enabled? boolean
        +++-rw authentication-key? string
        +++-rw crypto-algorithm identityref
---rw statistics
  +--ro state
    +--ro messages
      |  +--ro ack-sent?  yang:counter64
      |  +--ro ack-received?  yang:counter64
      |  +--ro bundle-sent?  yang:counter64
      |  +--ro bundle-received?  yang:counter64
      |  +--ro hello-sent?  yang:counter64
      |  +--ro hello-received?  yang:counter64
      |  +--ro integrity-challenge-sent?  yang:counter64
      |  +--ro integrity-challenge-received?  yang:counter64
      |  +--ro integrity-response-sent?  yang:counter64
      |  +--ro integrity-response-received?  yang:counter64
      |  +--ro notify-sent?  yang:counter64
      |  +--ro notify-received?  yang:counter64
      |  +--ro path-sent?  yang:counter64
      |  +--ro path-received?  yang:counter64
      |  +--ro path-err-sent?  yang:counter64
      |  +--ro path-err-received?  yang:counter64
      |  +--ro path-tear-sent?  yang:counter64
      |  +--ro path-tear-received?  yang:counter64
      |  +--ro resv-sent?  yang:counter64
      |  +--ro resv-received?  yang:counter64
      |  +--ro resv-confirm-sent?  yang:counter64
      |  +--ro resv-confirm-received?  yang:counter64
      |  +--ro resv-err-sent?  yang:counter64
      |  +--ro resv-err-received?  yang:counter64
      |  +--ro resv-tear-sent?  yang:counter64
      |  +--ro resv-tear-received?  yang:counter64
      |  +--ro summary-refresh-sent?  yang:counter64
      |  +--ro summary-refresh-received?  yang:counter64
      |  +--ro unknown-messages-received?  yang:counter64
    +--ro packets
      |  +--ro sent?  yang:counter64
      |  +--ro received?  yang:counter64
    +--ro errors
      |  +--ro authenticate?  yang:counter64
      |  +--ro checksum?  yang:counter64
      |  +--ro packet-len?  yang:counter64
  ---rw interface* [interface]
    +--rw interface  if:interface-ref
      +--rw refresh-reduction
        |  +--rw enabled?  boolean
      +--rw hellos
        |  +--rw enabled?  boolean
      +--rw authentication
        |  +--rw enabled?  boolean
        |  +--rw authentication-key?  string
|     |  +--rw crypto-algorithm      identityref
|     +--rw statistics
|     +--ro state
|     |     +--ro messages
|     |     |     +--ro ack-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro ack-received?
|     |     |     |       yang:counter64
|     |     |     +--ro bundle-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro bundle-received?
|     |     |     |       yang:counter64
|     |     |     +--ro hello-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro hello-received?
|     |     |     |       yang:counter64
|     |     |     +--ro integrity-challenge-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro integrity-challenge-received?
|     |     |     |       yang:counter64
|     |     |     +--ro integrity-response-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro integrity-response-received?
|     |     |     |       yang:counter64
|     |     |     +--ro notify-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro notify-received?
|     |     |     |       yang:counter64
|     |     |     +--ro path-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro path-received?
|     |     |     |       yang:counter64
|     |     |     +--ro path-err-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro path-err-received?
|     |     |     |       yang:counter64
|     |     |     +--ro path-tear-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro path-tear-received?
|     |     |     |       yang:counter64
|     |     |     +--ro resv-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro resv-received?
|     |     |     |       yang:counter64
|     |     |     +--ro resv-confirm-sent?
|     |     |     |       yang:counter64
|     |     |     +--ro resv-confirm-received?
|     |     |     |       yang:counter64

Figure 3: RSVP model tree diagram
2.3.2. YANG Module

```yamls
<CODE BEGINS> file "ietf-rsvp@2019-02-18.yang"
module ietf-rsvp {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp";
  /* Replace with IANA when assigned */
  prefix "rsvp";

  import ietf-interfaces {
    prefix if;
    reference "RFC8343: A YANG Data Model for Interface Management";
  }

  import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-routing {
    prefix "rt";
    reference "RFC8349: A YANG Data Model for Routing Management (NMDA Version)";
  }

  import ietf-key-chain {
    prefix "key-chain";
    reference "RFC8177: YANG Data Model for Key Chains";
  }

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

  contact
    "WG Web: <http://tools.ietf.org/wg/teas/>
      WG List: <mailto:teas@ietf.org>
    WG Chair: Lou Berger
      <mailto:lberger@labn.net>
```

description
"This module contains the RSVP YANG data model. The model fully conforms to the Network Management Datastore Architecture (NMDA).

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (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."

// RFC Ed.: replace XXXX with actual RFC number and remove this note.

// RFC Ed.: update the date below with the date of RFC publication // and remove this note.

revision "2019-02-18" {
  description
    "A YANG Data Model for Resource Reservation Protocol";
  reference
identity rsvp {
    base "rt:routing-protocol";
    description "RSVP protocol";
}

identity rsvp-session-type {
    description "Base RSVP session type";
}

identity rsvp-session-ipv4 {
    base rsvp-session-type;
    description "RSVP IPv4 session type";
}

identity rsvp-session-ipv6 {
    base rsvp-session-type;
    description "RSVP IPv4 session type";
}

identity reservation-style {
    description "Base identity for reservation style";
}

identity reservation-wildcard-filter {
    base reservation-style;
    description "Wildcard-Filter (WF) Style";
    reference "RFC2205";
}

identity reservation-fixed-filter {
    base reservation-style;
    description "Fixed-Filter (FF) Style";
    reference "RFC2205";
}

identity reservation-shared-explicit {
    base reservation-style;
    description "Shared Explicit (SE) Style";
    reference "RFC2205";
}

grouping graceful-restart-config {
    description "Base configuration parameters relating to RSVP"
grouping graceful-restart {
  description "RSVP graceful restart parameters grouping";
  container graceful-restart {
    description "RSVP graceful restart parameters container";
    uses graceful-restart-config;
  }
}

grouping refresh-reduction-config {
  description "Configuration parameters relating to RSVP refresh reduction";
  leaf enabled {
    type boolean;
    description "'true' if RSVP Refresh Reduction is enabled. 'false' if RSVP Refresh Reduction is disabled."
  }
}

grouping refresh-reduction {
  description "Top level grouping for RSVP refresh reduction parameters";
  container refresh-reduction {
    description "Top level container for RSVP refresh reduction parameters";
    uses refresh-reduction-config;
  }
}

grouping authentication-config {
  description "Configuration parameters relating to RSVP authentication";
}
leaf enabled {
    type boolean;
    description
    "'true' if RSVP Authentication is enabled. 'false' if RSVP Authentication is disabled.";
}
leaf authentication-key {
    type string;
    description
    "An authentication key string";
    reference
    "RFC 2747: RSVP Cryptographic Authentication";
}
leaf crypto-algorithm {
    type identityref {
        base key-chain:crypto-algorithm;
    }
    mandatory true;
    description
    "Cryptographic algorithm associated with key.";
}

grouping authentication {
    description
    "Top level grouping for RSVP authentication parameters";
    container authentication {
        description
        "Top level container for RSVP authentication parameters";
        uses authentication-config;
    }
}

grouping hellos-config {
    description
    "Configuration parameters relating to RSVP hellos";
    leaf enabled {
        type boolean;
        description
        "'true' if RSVP Hello is enabled. 'false' if RSVP Hello is disabled.";
    }
}

grouping hellos {
    description

"Top level grouping for RSVP hellos parameters";
canterior hellos {
    description
    "Top level container for RSVP hello parameters";
    uses hellos-config;
}
}

grouping signaling-parameters-config {
    description
    "Configuration parameters relating to RSVP signaling";
}

grouping signaling-parameters {
    description
    "Top level grouping for RSVP signaling parameters";
    uses signaling-parameters-config;
}

grouping session-attributes-state {
    description
    "Top level grouping for RSVP session properties";
    leaf local-index {
        type uint64;
        description
        "The index used to identify the RSVP session on the local network element. This index is generated by the device and is unique only to the local network element.";
    }
    leaf destination-port {
        type inet:port-number;
        description "RSVP destination port";
        reference "RFC2205";
    }
    leaf source {
        type inet:ip-address;
        description "RSVP source address";
        reference "RFC2205";
    }
    leaf destination {
        type inet:ip-address;
        description "RSVP destination address";
        reference "RFC2205";
    }
    leaf session-name {
        type string;
description "The signaled name of this RSVP session."
}
leaf session-state {
  type enumeration {
    enum "up" {
      description "RSVP session is up"
    }
    enum "down" {
      description "RSVP session is down"
    }
  }
  description "Enumeration of RSVP session states"
}
leaf session-type {
  type identityref {
    base rsvp-session-type;
  }
  description "RSVP session type"
}
container psbs {
  description "Path State Block container"
  list psb {
    description "List of path state blocks"
    leaf source-port {
      type inet:port-number;
      description "RSVP source port"
    }
    leaf expires-in {
      type uint32;
      units seconds;
      description "Time to reservation expiry (in seconds)"
    }
  }
}
container rsbs {
  description "Reservation State Block container"
  list rsb {
    description "List of reservation state blocks"
    leaf source-port {
      type inet:port-number;
      description "RSVP source port"
    }
  }
}
leaf reservation-style {
    type identityref {
        base reservation-style;
    }
    description "RSVP reservation style";
}
leaf expires-in {
    type uint32;
    units seconds;
    description "Time to reservation expiry (in seconds)";
}
}
}

grouping neighbor-attributes {
    description
        "Top level grouping for RSVP neighbor properties";
    leaf address {
        type inet:ip-address;
        description
            "Address of RSVP neighbor";
    }
    container state {
        config false;
        description
            "State information associated with RSVP neighbor properties";
        uses neighbor-derived-state;
    }
}

grouping packets-state {
    description
        "Packet statistics grouping";
    container packets {
        description
            "Packet statistics container";
        leaf sent {
            type yang:counter64;
            description
                "Packet sent count";
        }

        leaf received {
            type yang:counter64;
            description
                "Packet received count";
        }
    }
}
grouping protocol-state {
  description "RSVP protocol statistics grouping";
  container messages {
    description "RSVP protocol statistics container";
    leaf ack-sent {
      type yang:counter64;
      description "Hello sent count";
    }
  }
  leaf ack-received {
    type yang:counter64;
    description "Hello received count";
  }
  leaf bundle-sent {
    type yang:counter64;
    description "Bundle sent count";
  }
  leaf bundle-received {
    type yang:counter64;
    description "Bundle received count";
  }
  leaf hello-sent {
    type yang:counter64;
    description "Hello sent count";
  }
  leaf hello-received {
    type yang:counter64;
    description "Hello received count";
  }
  leaf integrity-challenge-sent {
    type yang:counter64;
  }
}
leaf integrity-challenge-received {
  type yang:counter64;
  description
    "Integrity Challenge received count";
}

leaf integrity-response-sent {
  type yang:counter64;
  description
    "Integrity Response sent count";
}

leaf integrity-response-received {
  type yang:counter64;
  description
    "Integrity Response received count";
}

leaf notify-sent {
  type yang:counter64;
  description
    "Notify sent count";
}

leaf notify-received {
  type yang:counter64;
  description
    "Notify received count";
}

leaf path-sent {
  type yang:counter64;
  description
    "Path sent count";
}

leaf path-received {
  type yang:counter64;
  description
    "Path received count";
}

leaf path-err-sent {
  type yang:counter64;
leaf path-err-sent {
    type yang:counter64;
    description
    "Path error sent count";
}

leaf path-err-received {
    type yang:counter64;
    description
    "Path error received count";
}

leaf path-tear-sent {
    type yang:counter64;
    description
    "Path tear sent count";
}

leaf path-tear-received {
    type yang:counter64;
    description
    "Path tear received count";
}

leaf resv-sent {
    type yang:counter64;
    description
    "Resv sent count";
}

leaf resv-received {
    type yang:counter64;
    description
    "Resv received count";
}

leaf resv-confirm-sent {
    type yang:counter64;
    description
    "Confirm sent count";
}

leaf resv-confirm-received {
    type yang:counter64;
    description
    "Confirm received count";
}

leaf resv-err-sent {
    type yang:counter64;
}
description
   "Resv error sent count";
}

leaf resv-err-received {
    type yang:counter64;
    description
       "Resv error received count";
}

leaf resv-tear-sent {
    type yang:counter64;
    description
       "Resv tear sent count";
}

leaf resv-tear-received {
    type yang:counter64;
    description
       "Resv tear received count";
}

leaf summary-refresh-sent {
    type yang:counter64;
    description
       "Summary refresh sent count";
}

leaf summary-refresh-received {
    type yang:counter64;
    description
       "Summary refresh received count";
}

leaf unknown-messages-received {
    type yang:counter64;
    description
       "Unknown packet received count";
}
}

grouping errors-state {
    description
       "Error statistics state grouping";
    container errors {
        description
           "Error statistics state container";
    }
}

leaf authenticate {
    type yang:counter64;
    description
    "The total number of packets received with an authentication failure.";
}

leaf checksum {
    type yang:counter64;
    description
    "The total number of packets received with an invalid checksum value.";
}

leaf packet-len {
    type yang:counter64;
    description
    "The total number of packets received with an invalid packet length.";
}
}
}

grouping statistics-state {
    description "RSVP statistic attributes.";
    container statistics {
        description
        "statistics state container";
        container state {
            config false;
            description
            "State information associated with RSVP hello parameters";
            uses protocol-state;
            uses packets-state;
            uses errors-state;
        }
    }
}

grouping neighbor-derived-state {
    description
    "Derived state at neighbor level.";

    leaf address {
        type inet:ip-address;
        description
        "Address of RSVP neighbor";
    }
}
leaf epoch {
    type uint32;
    description
        "Neighbor epoch.";
}

leaf expiry-time {
    type uint32;
    units seconds;
    description
        "Neighbor expiry time after which the neighbor state
         is purged if no states associated with it";
}

container graceful-restart {
    description
        "Graceful restart information.";

    leaf enabled {
        type boolean;
        description
            "'true' if graceful restart is enabled for the neighbor.";
    }

    leaf local-restart-time {
        type uint32;
        units seconds;
        description
            "Local node restart time";
    }

    leaf local-recovery-time {
        type uint32;
        units seconds;
        description
            "Local node recover time";
    }

    leaf neighbor-restart-time {
        type uint32;
        units seconds;
        description
            "Neighbor restart time";
    }

    leaf neighbor-recovery-time {
type uint32;
units seconds;
description
"Neighbor recover time";
}

container helper-mode {

description
"Helper mode information ";

leaf enabled {

type boolean;
description
"'true' if helper mode is enabled.";
}

leaf max-helper-restart-time {

type uint32;
units seconds;
description
"The time the router or switch waits after it discovers that a neighboring router has gone down before it declares the neighbor down";
}

leaf max-helper-recovery-time {

type uint32;
units seconds;
description
"The amount of time the router retains the state of its RSVP neighbors while they undergo a graceful restart";
}

leaf neighbor-restart-time-remaining {

type uint32;
units seconds;
description
"Number of seconds remaining for neighbor to send Hello message after restart.";
}

leaf neighbor-recovery-time-remaining {

type uint32;
units seconds;
description
"Number of seconds remaining for neighbor to refresh.";
}
leaf hello-status {
  type enumeration {
    enum "enabled" {
      description "Enabled";
    }
    enum "disabled" {
      description "Disabled";
    }
    enum "restarting" {
      description "Restarting";
    }
  }
  description "Hello status";
}

leaf interface {
  type if:interface-ref;
  description "Interface where RSVP neighbor was detected";
}

leaf neighbor-state {
  type enumeration {
    enum "up" {
      description "up";
    }
    enum "down" {
      description "down";
    }
    enum "hello-disable" {
      description "hello-disable";
    }
    enum "restarting" {
      description "restarting";
    }
  }
  description
"Neighbor state";
}

leaf refresh-reduction-capable {
  type boolean;
  description "enables all RSVP refresh reduction message
    bundling, RSVP message ID, reliable message delivery
    and summary refresh";
  reference
    "RFC 2961 RSVP Refresh Overhead Reduction
    Extensions";
}

leaf restart-count {
  type yang:counter32;
  description "Number of times this neighbor restart";
}

leaf restart-time {
  type yang:date-and-time;
  description "Last restart time of the neighbor";
}
}

grouping global-attributes {
  description "Top level grouping for RSVP global properties";
  container sessions {
    description "RSVP sessions container";
    list session {
      key "local-index";
      config false;
      description "List of RSVP sessions";
      leaf local-index {
        type leafref {
          path "/state/local-index";
        }
        description "Reference to the local index for the RSVP
          session";
      }
      container state {

config false;
description
"State information associated with RSVP
session parameters";
uses session-attributes-state;
}
}
}
uses statistics-state;
}
grouping intf-attributes {

description
"Top level grouping for RSVP interface properties";
uses signaling-parameters;
uses refresh-reduction;
uses hellos;
uses authentication;
uses statistics-state;
}
augment "/rt:routing/rt:control-plane-protocols/
+ "rt:control-plane-protocol" {
 when "rt:type = 'rsvp:rsvp'" {

description
"This augment is only valid when routing protocol
instance type is RSVP.";
}

description
"RSVP protocol augmentation";
container rsvp {
presence "Enable RSVP feature";
description "RSVP feature container";
container globals {

description "RSVP global properties.";
uses global-attributes;
uses graceful-restart;
}
}

container interfaces {

description
"RSVP interfaces container";
uses intf-attributes;

list interface {
 key "interface";

description
"RSVP interfaces.";
}
2.4. RSVP Extended YANG Model

The RSVP extended YANG model covers non-core RSVP feature(s). It also covers feature(s) that are not necessarily supported by all vendors, and hence, can be guarded with "if-feature" checks.

2.4.1. Tree Diagram

Figure 4 shows the YANG tree representation for configuration and state data that is augmenting the RSVP extended module:

```yang
module ietf-rsvp-extended
    augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
        /rsvp:graceful-restart:
            ++--rw restart-time?    uint32
            ++--rw recovery-time?   uint32
        augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
            /rsvp:statistics/rsvp:state/rsvp:packets:
                ++--ro discontinuity-time?   yang:date-and-time
                ++--ro out-dropped?          yang:counter64
                ++--ro in-dropped?           yang:counter64
                ++--ro out-error?            yang:counter64
                ++--ro in-error?             yang:counter64
```
/rt:control-plane-protocol/rsvp:globals
/rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:globals
/rt:routing/rt:control-plane-protocols
/augment /rt:routing/rt:control-plane-protocols

++--rw refresh-interval?        uint32
++--rw refresh-misses?          uint32
++--rw checksum?                boolean
++--rw patherr-state-removal?   empty
augment /rt:routing/rt:control-plane-protocols
/augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:refresh-reduction:
++--rw bundle-message-max-size?    uint32
++--rw reliable-ack-hold-time?     uint32
++--rw reliable-ack-max-size?      uint32
++--rw reliable-retransmit-time?   uint32
++--rw reliable-srefresh?          empty
++--rw summary-max-size?           uint32
augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:hellos:
++--rw interface-based?   empty
++--rw hello-interval?    uint32
++--rw hello-misses?      uint32
augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:authentication:
++--rw lifetime?      uint32
++--rw window-size?   uint32
++--rw challenge?     empty
++--rw retransmits?   uint32
++--rw key-chain?     key-chain:key-chain-ref
augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:interface:
++--rw refresh-interval?        uint32
++--rw refresh-misses?          uint32
++--rw checksum?                boolean
++--rw patherr-state-removal?   empty
augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/rsvp:interface/refresh-reduction:
++--rw bundle-message-max-size?    uint32
++--rw reliable-ack-hold-time?     uint32
++--rw reliable-ack-max-size?      uint32
++--rw reliable-retransmit-time?   uint32
2.4.2. YANG Module

Figure 5 shows the RSVP extended YANG module:

```yang
module ietf-rsvp-extended {
    yang-version 1.1;
    prefix "rsvp-ext";

    import ietf-rsvp {
        prefix "rsvp";
        reference
            "RFCXXXX: A YANG Data Model for Resource Reservation Protocol (RSVP)";
    }

    import ietf-routing {
        prefix "rt";
        reference
            "RFC8349: A YANG Data Model for Routing Management (NMDA Version)";
    }

    import ietf-yang-types {
        prefix "yang";
        reference "RFC6991: Common YANG Data Types";
    }
}
```

Figure 4: RSVP extended model tree diagram
import ietf-key-chain {
    prefix "key-chain";
    reference "RFC8177: YANG Data Model for Key Chains";
}

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

contact
    "WG Web:  <http://tools.ietf.org/wg/teas/>
    WG List:  <mailto:teas@ietf.org>
    WG Chair: Lou Berger
               <mailto:lberger@labn.net>
    WG Chair: Vishnu Pavan Beeram
               <mailto:vbeeram@juniper.net>
    Editor:  Vishnu Pavan Beeram
              <mailto:vbeeram@juniper.net>
    Editor:  Tarek Saad
              <mailto:tsaad@cisco.com>
    Editor:  Rakesh Gandhi
              <mailto:rgandhi@cisco.com>
    Editor:  Himanshu Shah
              <mailto:hshah@ciena.com>
    Editor:  Xufeng Liu
              <mailto:Xufeng_Liu@jabil.com>
    Editor:  Xia Chen
              <mailto:jescia.chenxia@huawei.com>
    Editor:  Raqib Jones
              <mailto:raqib@Brocade.com>
    Editor:  Bin Wen
              <mailto:Bin_Wen@cable.comcast.com>"

description
    "This module contains the Extended RSVP YANG data model."
The model fully conforms to the Network Management Datastore Architecture (NMDA).

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (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."

// RFC Ed.: replace XXXX with actual RFC number and remove this note.

// RFC Ed.: update the date below with the date of RFC publication and remove this note.

revision "2019-02-18" {
  description
    "A YANG Data Model for Extended Resource Reservation Protocol";
  reference
    "RFCXXXX: A YANG Data Model for Extended Resource Reservation Protocol (RSVP)";
}

/* RSVP features */

feature authentication {
  description
    "Indicates support for RSVP authentication";
}

feature error-statistics {
  description
    "Indicates support for error statistics";
}

feature global-statistics {
  description
    "Indicates support for global statistics";
}

feature graceful-restart {
  description

"Indicates support for RSVP graceful restart";
}

feature hellos {
  description
    "Indicates support for RSVP hellos (RFC3209).";
}

feature notify {
  description
    "Indicates support for RSVP notify message (RFC3473).";
}

feature refresh-reduction {
  description
    "Indicates support for RSVP refresh reduction (RFC2961).";
}

feature refresh-reduction-extended {
  description
    "Indicates support for RSVP refresh reduction (RFC2961).";
}

feature per-interface-statistics {
  description
    "Indicates support for per interface statistics";
}

grouping graceful-restart-extended-config {
  description
    "Configuration parameters relating to RSVP Graceful-Restart";
  leaf restart-time {
    type uint32;
    units seconds;
    description
    "Graceful restart time (seconds).";
    reference
  }
  leaf recovery-time {
    type uint32;
    units seconds;
    description
    "RSVP state recovery time";
  }
}
grouping authentication-extended-config {
  description
    "Configuration parameters relating to RSVP authentication";
  leaf lifetime {
    type uint32 {
      range "30..86400";
    }
    units seconds;
    description
      "Life time for each security association";
    reference
      "RFC 2747: RSVP Cryptographic Authentication";
  }
  leaf window-size {
    type uint32 {
      range "1..64";
    }
    description
      "Window-size to limit number of out-of-order messages.";
    reference
      "RFC 2747: RSVP Cryptographic Authentication";
  }
  leaf challenge {
    type empty;
    description
      "Enable challenge messages.";
    reference
      "RFC 2747: RSVP Cryptographic Authentication";
  }
  leaf retransmits {
    type uint32 {
      range "1..10000";
    }
    description
      "Number of retransmits when messages are dropped.";
    reference
      "RFC 2747: RSVP Cryptographic Authentication";
  }
  leaf key-chain {
grouping signaling-parameters-extended-config {

    leaf hello-misses {
        type uint32 {
            range "1..10";
        }
        description
            "Configure max number of consecutive missed
            Hello messages.";
        reference
            "RFC 3209: RSVP-TE: Extensions to RSVP for
            LSP Tunnels RFC 5495: Description of the
            Resource Reservation Protocol - Traffic-Engineered
            (RSVP-TE) Graceful Restart Procedures";
    }
}

}
description
   "Configuration parameters relating to RSVP signaling";
leaf refresh-interval {
    type uint32;
    description
       "Set interval between successive refreshes";
}
leaf refresh-misses {
    type uint32;
    description
       "Set max number of consecutive missed messages for state expiry";
}
leaf checksum {
    type boolean;
    description
       "Enable RSVP message checksum computation";
}
leaf patherr-state-removal {
    type empty;
    description
       "State-Removal flag in Path Error message if present.";
}
}

grouping refresh-reduction-extended-config {
    description
       "Configuration parameters relating to RSVP refresh reduction";

    leaf bundle-message-max-size {
        type uint32 {
            range "512..65000";
        }
        description
           "Configure maximum size (bytes) of a single RSVP Bundle message.";
    }
    leaf reliable-ack-hold-time {
        type uint32;
        units milliseconds;
        description
           "Configure hold time in milliseconds for sending RSVP ACK message(s).";
    }
    leaf reliable-ack-max-size {

type uint32;
description
  "Configure max size of a single RSVP ACK
  message.";
}
leaf reliable-retransmit-time {
  type uint32;
  units milliseconds;
  description
  "Configure min delay in milliseconds to
  wait for an ACK before a retransmit.";
}
leaf reliable-srefresh {
  type empty;
  description
  "Configure use of reliable messaging for
  summary refresh if present.";
}
leaf summary-max-size {
  type uint32 {
    range "20..65000";
  }
  description
  "Configure max size (bytes) of a single
  RSVP summary refresh message.";
}
}
}

grouping packets-extended-state {
  description
  "Packet statistics.";
  leaf discontinuity-time {
    type yang:date-and-time;
    description
    "The time on the most recent occasion at which any one
    or more of the statistic counters suffered a
    discontinuity. If no such discontinuities have occurred
    since the last re-initialization of the local
    management subsystem, then this node contains the time
    the local management subsystem re-initialized itself.";
  }
  leaf out-dropped {
    type yang:counter64;
    description
    "Out packet drop count";
  }
  leaf in-dropped {


```yang

type yang:counter64;
description "In packet drop count";
}

leaf out-error {
type yang:counter64;
description "Out packet error count";
}

leaf in-error {
type yang:counter64;
description "In packet rx error count";
}
}

grouping protocol-extended-state {
description "RSVP protocol statistics.";
}

grouping errors-extended-state {
description "Error statistics.";
}

grouping extended-state {
description "RSVP statistic attributes.";
uses packets-extended-state;
uses protocol-extended-state;
uses errors-extended-state;
}

/**
 * RSVP extensions augmentations
 */

/* RSVP globals graceful restart*/
augment "/rt:routing/rt:control-plane-protocols/" +
   "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" +
   "rsvp:graceful-restart" {
description "RSVP globals configuration extensions";
uses graceful-restart-extended-config;
}

/* RSVP statistics augmentation */
```
augment "rt:routing/rt:control-plane-protocols/" +
"rt:control-plane-protocol/rsvp/rsvp:globals/" +
"rsvp:statistics/rsvp:state/rsvp:packets" {
  description
  "RSVP packet stats extensions";
  uses packets-extended-state;
}

augment "rt:routing/rt:control-plane-protocols/" +
"rt:control-plane-protocol/rsvp/rsvp:globals/" +
"rsvp:statistics/rsvp:state/rsvp:messages" {
  description
  "RSVP protocol message stats extensions";
  uses protocol-extended-state;
}

augment "rt:routing/rt:control-plane-protocols/" +
"rt:control-plane-protocol/rsvp/rsvp:globals/" +
"rsvp:statistics/rsvp:state/rsvp:errors" {
  description
  "RSVP errors stats extensions";
  uses errors-extended-state;
}

/**
 * RSVP all interfaces extensions
 */

/* RSVP interface signaling extensions */
augment "rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp/rsvp:interfaces" {
    description
    "RSVP signaling all interfaces configuration extensions";
    uses signaling-parameters-extended-config;
}

/* RSVP refresh reduction extension */
augment "rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp/rsvp:interfaces/"
  + "rsvp:refresh-reduction" {
    description
    "RSVP refresh-reduction all interface configuration
     extensions";
    uses refresh-reduction-extended-config;
}

/* RSVP hellos extension */
augment "rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp/rsvp:interfaces/"
  + "rsvp:hellos" {

description
  "RSVP hello all interfaces configuration extensions";
  uses hellos-extended-config;
}

/* RSVP authentication extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:authentication" {
    description
    "RSVP authentication all interfaces configuration extensions";
    uses authentication-extended-config;
  }

/**
 * RSVP interface extensions
 */

/* RSVP interface signaling extensions */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface" {
    description
    "RSVP signaling interface configuration extensions";
    uses signaling-parameters-extended-config;
  }

/* RSVP refresh reduction extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface/rsvp:refresh-reduction" {
    description
    "RSVP refresh-reduction interface configuration extensions";
    uses refresh-reduction-extended-config;
  }

/* RSVP hellos extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface/rsvp:hellos" {
    description
    "RSVP hello interface configuration extensions";
    uses hellos-extended-config;
  }

/* RSVP authentication extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/" 
    "rsvp:authentication" {
    description
    "RSVP authentication all interfaces configuration extensions";
    uses authentication-extended-config;
  }

/**
 * RSVP interface extensions
 */

/* RSVP interface signaling extensions */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface" {
    description
    "RSVP signaling interface configuration extensions";
    uses signaling-parameters-extended-config;
  }

/* RSVP refresh reduction extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface/rsvp:refresh-reduction" {
    description
    "RSVP refresh-reduction interface configuration extensions";
    uses refresh-reduction-extended-config;
  }

/* RSVP hellos extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/
    "rsvp:interface/rsvp:hellos" {
    description
    "RSVP hello interface configuration extensions";
    uses hellos-extended-config;
  }

/* RSVP authentication extension */
augment "/rt:routing/rt:control-plane-protocols/
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/" 

"rsvp:interface/rsvp:authentication" {
  description
    "RSVP authentication interface configuration extensions";
  uses authentication-extended-config;
}

Figure 5: RSVP extended YANG module

3. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

XML: N/A, the requested URI is an XML namespace.

XML: N/A, the requested URI is an XML namespace.

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

name:       ietf-rsvp
prefix:     ietf-rsvp
reference:  RFCXXXX

name:       ietf-rsvp-extended
prefix:     ietf-rsvp-extended
reference:  RFCXXXX

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

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., `<edit-config>`) to these data nodes without proper protection can have a negative effect on network operations.

5. Acknowledgement

The authors would like to thank Lou Berger for reviewing and providing valuable feedback on this document.

6. Contributors

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones
Brocade

Email: raqib@Brocade.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

7. Normative References

[I-D.ietf-teas-yang-rsvp-te]


Authors' Addresses

Vishnu Pavan Beeram
Juniper Networks
Email: vbeeram@juniper.net

Tarek Saad (editor)
Cisco Systems, Inc.
Email: tsaad@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Email: rgandhi@cisco.com

Xufeng Liu
Jabil
Email: Xufeng_Liu@jabil.com
A YANG Data Model for RSVP-TE Protocol

draft-ietf-teas-yang-rsvp-te-06

Abstract

This document defines a YANG data model for the configuration and management of RSVP (Resource Reservation Protocol) to establish Traffic-Engineered (TE) Label-Switched Paths (LSPs) for MPLS (Multi-Protocol Label Switching) and other technologies.

The model defines a generic RSVP-TE module for signaling LSPs that is technology agnostic. The generic RSVP-TE module is to be augmented by technology specific RSVP-TE modules that define technology specific data. This document also defines the augmentation for RSVP-TE MPLS LSPs model.

This model covers data for the configuration, operational state, remote procedural calls, and event notifications.

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."
1. Introduction

YANG [RFC7950] is a data modeling language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG has proved relevant beyond its initial confines, as bindings to other interfaces (e.g. RESTCONF [RFC8040]) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the
This document defines a generic YANG data model for configuring and managing RSVP-TE LSP(s) [RFC3209]. The RSVP-TE generic model augments the RSVP base and extended models defined in [I-D.ietf-teas-yang-rsvp], and adds TE extensions to the RSVP protocol [RFC2205] model configuration and state data. The technology specific RSVP-TE models augment the generic RSVP-TE model with additional technology specific parameters. For example, this document also defines the MPLS RSVP-TE model for configuring and managing MPLS RSVP TE LSP(s).

In addition to augmenting the RSVP YANG module, the modules defined in this document augment the TE Interfaces, Tunnels and LSP(s) YANG module defined in [I-D.ietf-teas-yang-te] to define additional parameters to enable signaling for RSVP-TE.

1.1. Terminology

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

The terminology for describing YANG data models is found in [RFC7950].

1.2. Prefixes in Data Node Names

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

The RSVP-TE generic model augments the RSVP base and extended YANG models defined in [I-D.ietf-teas-yang-rsvp]. It also augments the TE tunnels and interfaces module defined in [I-D.ietf-teas-yang-te] to cover parameters specific to the configuration and management of RSVP-TE interfaces, tunnels and LSP(s).

The RSVP-TE MPLS YANG model augments the RSVP-TE generic model with parameters to configure and manage signaling of MPLS RSVP-TE LSPs. RSVP-TE model augmentation for other dataplane technologies (e.g. OTN or WDM) are outside the scope of this document.

There are three types of configuration and state data nodes in module(s) defined in this document:

- those augmenting or extending the base RSVP module that is defined in [I-D.ietf-teas-yang-rsvp]
- those augmenting or extending the base TE module defined in [I-D.ietf-teas-yang-te]
- those that are specific to the RSVP-TE and RSVP-TE MPLS modules defined in this document.

2.1. Module Relationship

The data pertaining to RSVP-TE in this document is divided into two modules: a technology agnostic RSVP-TE module that holds generic parameters for RSVP-TE applicable to all technologies, and a MPLS technology specific RSVP-TE module that holds parameters specific to MPLS technology.
The relationship between the different modules is shown in Figure 1.

```
TE basic       +---------+    o: augment
    module    | ietf-te |

+---------+    o

RSVP-TE module | ietf-rsvp-te | o . . .
               +---------+   \
                  o
RSVP module     | ietf-rsvp |
                +---------+  RSVP-TE with MPLS

RSVP extended module          +-------------------+
                             | ietf-rsvp-extended |
                             +-------------------+
```

Figure 1: Relationship of RSVP and RSVP-TE modules with other protocol modules

2.2. Model Tree Diagrams

A full tree diagram of the module(s) defined in this document as per the syntax defined in [RFC8340] are given in subsequent sections.

2.2.1. RSVP-TE Model Tree Diagram

Figure 2 shows the YANG tree diagram of the RSVP-TE generic YANG model defined in module ietf-rsvp-te.yang.

```
module: ietf-rsvp-te
    augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/rsvp:rsvp:rsvp:globals:
            +--rw global-soft-preemption!
            +--rw soft-preemption-timeout?  uint16
    augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/rsvp:rsvp:rsvp:interfaces:
            +--rw rsvp-te-interface-attributes
            +--ro state
    augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/rsvp:rsvp:rsvp:interfaces
            /rsvp:interface:
```

---rw rsvp-te-interface-attributes
  ---ro state
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
  /rsvp:sessions/rsvp:session/rsvp:state/rsvp:psbs/rsvp:psb:
  ---ro tspec-average-rate?    rt-types:bandwidth-ieee-float32
  ---ro tspec-size?            rt-types:bandwidth-ieee-float32
  ---ro tspec-peak-rate?       rt-types:bandwidth-ieee-float32
  ---ro min-policed-unit?      uint32
  ---ro max-packet-size?       uint32
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
  /rsvp:sessions/rsvp:session/rsvp:state/rsvp:rsbs/rsvp:rsb:
  ---ro fspec-average-rate?    rt-types:bandwidth-ieee-float32
  ---ro fspec-size?            rt-types:bandwidth-ieee-float32
  ---ro fspec-peak-rate?       rt-types:bandwidth-ieee-float32
  ---ro min-policed-unit?      uint32
  ---ro max-packet-size?       uint32
augment /rt:routing/rt:control-plane-protocols
  /rt:neighbors:
  augment /te:te/te:tunnels/te:tunnel:
    ---rw lsp-signaled-name?    string
    ---rw session-attribute*    identityref
    ---rw lsp-attribute*        identityref
    ---rw retry-timer?          uint16
augment /te:te/te:lsps-state/te:lsp:
  ---ro associated-rsvp-session?   leafref
  ---ro lsp-signaled-name?         string
  ---ro session-attribute*         identityref
  ---ro lsp-attribute*             identityref
  ---ro incoming-explicit-route-hop* [index]
    ---ro index                uint32
    ---ro (type)?
      ++-(numbered-node-hop)
        ---ro numbered-node-hop
        ---ro node-id    te-node-id
        ---ro hop-type?  te-hop-type
      ++-(numbered-link-hop)
        ---ro numbered-link-hop
        ---ro link-tp-id  te-tp-id
        ---ro hop-type?  te-hop-type
        ---ro direction? te-link-direction
      ++-(unnumbered-link-hop)
        ---ro unnumbered-link-hop
        ---ro link-tp-id  te-tp-id
        ---ro node-id    te-node-id
        ---ro hop-type?  te-hop-type
+--ro flags* path-attribute-flags
++--:(numbered-link-hop)
  +--ro numbered-link-hop
    +--ro link-tp-id  te-tp-id
    +--ro flags*  path-attribute-flags
++--:(unnumbered-link-hop)
  +--ro unnumbered-link-hop
    +--ro link-tp-id  te-tp-id
    +--ro node-id?  te-node-id
    +--ro flags*  path-attribute-flags
++--:(label)
  +--ro label-hop
    +--ro te-label
    |  +--ro (technology)?
    |    |  +--:(generic)
    |    |     +--ro generic?  rt-types:generalized-label
    |    +--ro direction?  te-label-direction
    +--ro flags*  path-attribute-flags
++--ro outgoing-record-route-subobjects
  +--ro outgoing-record-route-subobject* [index]
    +--ro index  uint32
    +--ro (type)?
      |  +--ro outgoing-record-route-subobject
      |     +--ro numbered-node-hop
      |        +--ro node-id  te-node-id
      |        +--ro flags*  path-attribute-flags
      +--ro numbered-link-hop
        +--ro link-tp-id  te-tp-id
        +--ro flags*  path-attribute-flags
      +--ro unnumbered-link-hop
        +--ro link-tp-id  te-tp-id
        +--ro node-id?  te-node-id
        +--ro flags*  path-attribute-flags
      +--ro label
        +--ro label-hop
        |  +--ro te-label
        |    |  +--ro (technology)?
        |    |    |  +--:(generic)
        |    |    |     +--ro generic?  rt-types:generalized-label
        |    |    +--ro direction?  te-label-direction
        |    +--ro flags*  path-attribute-flags
augment /te:te/te:tunnels/te:tunnel/te:p2p-primary-paths
  /te:p2p-primary-path/te:lsps/te:lsp:
    +--ro associated-rsvp-session?  leafref
    +--ro lsp-signaled-name?  string
    +--ro session-attribute*  identityref
+++ ro lsp-attribute* identityref
+++ ro explicit-route-objects
    +++ ro incoming-explicit-route-hop* [index]
    +++ ro index uint32
    +++ ro (type)?
      +++:(numbered-node-hop)
        +++ ro numbered-node-hop
          +++ ro node-id te-node-id
          +++ ro hop-type? te-hop-type
      +++:(numbered-link-hop)
        +++ ro numbered-link-hop
          +++ ro link-tp-id te-tp-id
          +++ ro hop-type? te-hop-type
          +++ ro direction? te-link-direction
      +++:(unnumbered-link-hop)
        +++ ro unnumbered-link-hop
          +++ ro link-tp-id te-tp-id
          +++ ro node-id te-node-id
          +++ ro hop-type? te-hop-type
          +++ ro direction? te-link-direction
      +++:(as-number)
        +++ ro as-number-hop
          +++ ro as-number inet:as-number
          +++ ro hop-type? te-hop-type
      +++:(label)
        +++ ro label-hop
          +++ ro te-label
            +++ ro (technology)?
              +++:(generic)
                +++ ro generic? rt-types:generalized-label
                +++ ro direction? te-label-direction
    +++ ro outgoing-explicit-route-hop* [index]
    +++ ro index uint32
    +++ ro (type)?
      +++:(numbered-node-hop)
        +++ ro numbered-node-hop
          +++ ro node-id te-node-id
          +++ ro hop-type? te-hop-type
      +++:(numbered-link-hop)
        +++ ro numbered-link-hop
          +++ ro link-tp-id te-tp-id
          +++ ro hop-type? te-hop-type
          +++ ro direction? te-link-direction
      +++:(unnumbered-link-hop)
        +++ ro unnumbered-link-hop
          +++ ro link-tp-id te-tp-id
          +++ ro node-id te-node-id
          +++ ro hop-type? te-hop-type
---ro direction?  te-link-direction
---:(as-number)
  --ro as-number-hop
    ---ro as-number  inet:as-number
    ---ro hop-type?  te-hop-type
---:(label)
  --ro label-hop
    --->ro te-label
      ---ro (technology)?
        ---:(generic)
          ---ro generic?  rt-types:generalized-label
          ---ro direction?  te-label-direction
---ro incoming-record-route-subobjects
  ---ro incoming-record-route-subobject* [index]
    ---ro index  uint32
    ---ro (type)?
      ---:(numbered-node-hop)
        --ro numbered-node-hop
          ---ro node-id  te-node-id
          ---ro flags*  path-attribute-flags
      ---:(numbered-link-hop)
        --ro numbered-link-hop
          ---ro link-tp-id  te-tp-id
          ---ro flags*  path-attribute-flags
      ---:(unnumbered-link-hop)
        --ro unnumbered-link-hop
          ---ro link-tp-id  te-tp-id
          ---ro node-id?  te-node-id
          ---ro flags*  path-attribute-flags
      ---:(label)
        --ro label-hop
          --->ro te-label
            ---ro (technology)?
              ---:(generic)
                ---ro generic?  rt-types:generalized-label
                ---ro direction?  te-label-direction
          ---ro flags*  path-attribute-flags
---ro outgoing-record-route-subobjects
  ---ro outgoing-record-route-subobject* [index]
    ---ro index  uint32
    ---ro (type)?
      ---:(numbered-node-hop)
        --ro numbered-node-hop
          ---ro node-id  te-node-id
          ---ro flags*  path-attribute-flags
      ---:(numbered-link-hop)
        --ro numbered-link-hop
          ---ro link-tp-id  te-tp-id
|    +--ro flags*       path-attribute-flags
|    +--:(unnumbered-link-hop)
|    |    +--ro unnumbered-link-hop
|    |    |    +--ro link-tp-id   te-tp-id
|    |    |    +--ro node-id?  te-node-id
|    |    +--ro flags*       path-attribute-flags
|    +--:(label)
|    |    +--ro label-hop
|    |    |    +--ro te-label
|    |    |    |    +--ro (technology)?
|    |    |    |    |    +--ro generic?   rt-types:generalized-label
|    |    |    |    +--ro direction?  te-label-direction
|    |    +--ro flags*       path-attribute-flags
|    augment /te:te/te:tunnels/te:tunnel/te:p2p-primary-paths
|    /te:p2p-primary-path/te:lsp-provisioning-error-infos
|    /te:lsp-provisioning-error-info:
|    |    +--ro rsvp-message-type?  identityref
|    |    +--ro rsvp-error-code?   uint8
|    |    +--ro rsvp-error-subcode? uint16
|    augment /te:te/te:tunnels/te:tunnel/te:p2p-primary-paths
|    /te:p2p-primary-path/te:lsp/te:lsp-provisioning-error-infos
|    /te:lsp-provisioning-error-info:
|    |    +--ro rsvp-message-type?  identityref
|    |    +--ro rsvp-error-code?   uint8
|    |    +--ro rsvp-error-subcode? uint16
|    augment /te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths
|    /te:p2p-secondary-path/te:lsp/te:lsp:
|    |    +--ro associated-rsvp-session? leafref
|    |    +--ro lsp-signaled-name?  string
|    |    +--ro session-attribute*  identityref
|    |    +--ro lsp-attribute*      identityref
|    +--ro explicit-route-objects
|    |    +--ro incoming-explicit-route-hop* [index]
|    |    |    +--ro index               uint32
|    |    |    +--ro (type)?
|    |    |    |    +--:(numbered-node-hop)
|    |    |    |    |    +--ro node-id    te-node-id
|    |    |    |    |    +--ro hop-type?  te-hop-type
|    |    |    |    +--:(numbered-link-hop)
|    |    |    |    |    +--ro link-tp-id   te-tp-id
|    |    |    |    |    +--ro hop-type?  te-hop-type
|    |    |    |    |    +--ro direction?  te-link-direction
|    |    |    |    +--:(unnumbered-link-hop)
|    |    |    +--ro unnumbered-link-hop
++ro link-tp-id    te-tp-id
++ro node-id      te-node-id
++ro hop-type?    te-hop-type
++ro direction?   te-link-direction
+++:(as-number)
  +++ro as-number-hop
    +++ro as-number    inet:as-number
    +++ro hop-type?    te-hop-type
+++:(label)
  +++ro label-hop
    +++ro te-label
      +++ro (technology)?
        +++:(generic)
          +++ro generic?   rt-types:generalized-label
          +++ro direction?   te-label-direction
  +++ro outgoing-explicit-route-hop* [index]
    +++ro index                        uint32
    +++ro (type)?
      +++:(numbered-node-hop)
        +++ro numbered-node-hop
          +++ro node-id      te-node-id
          +++ro hop-type?    te-hop-type
      +++:(numbered-link-hop)
        +++ro numbered-link-hop
          +++ro link-tp-id    te-tp-id
          +++ro hop-type?    te-hop-type
          +++ro direction?   te-link-direction
      +++:(unnumbered-link-hop)
        +++ro unnumbered-link-hop
          +++ro link-tp-id    te-tp-id
          +++ro node-id      te-node-id
          +++ro hop-type?    te-hop-type
          +++ro direction?   te-link-direction
+++:(as-number)
  +++ro as-number-hop
    +++ro as-number    inet:as-number
    +++ro hop-type?    te-hop-type
+++:(label)
  +++ro label-hop
    +++ro te-label
      +++ro (technology)?
        +++:(generic)
          +++ro generic?   rt-types:generalized-label
          +++ro direction?   te-label-direction
+++ro incoming-record-route-subobjects
  +++ro incoming-record-route-subobject* [index]
    +++ro index                        uint32
    +++ro (type)?
++--:(numbered-node-hop)
  |   +--:(numbered-node-hop)
  |      |   +--ro numbered-node-hop
  |      |     |   +--ro node-id    te-node-id
  |      |     |   +--ro flags*     path-attribute-flags
  |      +--:(numbered-link-hop)
  |          |   +--ro numbered-link-hop
  |          |     |   +--ro link-tp-id    te-tp-id
  |          |     |   +--ro flags*        path-attribute-flags
  |          +--:(unnumbered-link-hop)
  |              +--ro unnumbered-link-hop
  |              |   +--ro link-tp-id    te-tp-id
  |              |   +--ro node-id?      te-node-id
  |              |   +--ro flags*        path-attribute-flags
  |              +--:(label)
  |                 |   +--ro label-hop
  |                 |      |   +--ro te-label
  |                 |      |      |   +--ro (technology)?
  |                 |      |      +--:(generic)
  |                 |      |      |   +--ro generic?   rt-types:generalized-label
  |                 |      |      |   +--ro direction?       te-label-direction
  |                 |      |   +--ro flags*      path-attribute-flags
  |                 +--ro outgoing-record-route-subobjects
  |                 |   +--ro outgoing-record-route-subobject* [index]
  |                 |       |   +--ro index            uint32
  |                 |       |   +--ro (type)?
  |                 +--:(numbered-node-hop)
  |                      |   +--ro numbered-node-hop
  |                      |      |   +--ro node-id    te-node-id
  |                      |      |   +--ro flags*     path-attribute-flags
  |                      +--:(numbered-link-hop)
  |                           +--ro numbered-link-hop
  |                           |   +--ro link-tp-id    te-tp-id
  |                           |   +--ro flags*        path-attribute-flags
  |                           +--:(unnumbered-link-hop)
  |                               +--ro unnumbered-link-hop
  |                               |   +--ro link-tp-id    te-tp-id
  |                               |   +--ro node-id?      te-node-id
  |                               |   +--ro flags*        path-attribute-flags
  |                               +--:(label)
  |                                  +--ro label-hop
  |                                     |   +--ro te-label
  |                                     |      |   +--ro (technology)?
  |                                     |      |      |   +--:(generic)
  |                                     |      |      |   |   +--ro generic?   rt-types:generalized-label
  |                                     |      |      |   |   +--ro direction?       te-label-direction
  |                                     |      |   +--ro flags*      path-attribute-flags
  |                                     +--ro outgoing-record-route-subobjects
augment /te:te/tetunnels/te:tunnel/te:p2p-secondary-paths
    /te:p2p-secondary-path/te:lsp-provisioning-error-infos
2.2.2. RSVP-TE MPLS Model Tree Diagram

Figure 5 shows the YANG tree diagram of the RSVP-TE MPLS YANG model defined in module ietf-rsvp-te-mpls.yang and that augments RSVP-TE module as well as RSVP and TE YANG modules.
---ro backup-tunnel-name?          string
---ro backup-frr-on?               uint8
---ro backup-protected-lsp-num?    uint32
augment /te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths
    /te:p2p-secondary-path/te:lsps/te:lsp:
    ---ro session-attribute*    identityref
---ro backup-info
    ---ro backup-tunnel-name?          string
    ---ro backup-frr-on?               uint8
    ---ro backup-protected-lsp-num?    uint32
augment /te:te/te-dev:interfaces/te-dev:interface:
    +++rw bandwidth-mpls-reservable
        +++rw (bandwidth-value)?
            | ---rw absolute-value?          te-packet-types:bandwidth-kbps
            | ---rw (percentage)
            |     | ---rw percent-value?          uint32
        +++rw (bc-model-type)?
            | ---:(bc-model-rdm)
                | ---rw (absolute)
                | ---rw absolute-value?          te-packet-types:bandwidth-kbps
                | ---rw (percentage)
                |     | ---rw percent-value?          uint32
            | ---:(bc-model-mam)
                | ---rw (bc-model-mam)
                | ---rw bandwidth-mpls-constraints
                    | ---rw maximum-reservable?
                    |     |     te-packet-types:bandwidth-kbps
                    |     | ---rw bc-value*               uint32
            | ---:(bc-model-mar)
                | ---rw (bc-model-mar)
                | ---rw bandwidth-mpls-constraints
                    | ---rw maximum-reservable?
                    |     |     te-packet-types:bandwidth-kbps
                    |     | ---rw bc-value*               uint32
augment /te:te/te-dev:interfaces/te-dev:interface:
    +++rw rsvp-te-frr-backups
        +++rw (type)?
            | ---(static-tunnel)
                | ---rw static-backups
                | ---rw static-backup* [backup-tunnel-name]
                |     |     -> /te:te/tunnels/tunnel/name
            | ---(auto-tunnel)
                | ---rw auto-tunnel-backups
                | ---rw auto-backup-protection?    identityref
                | ---rw auto-backup-path-computation? identityref
2.3. YANG Modules

2.3.1. RSVP-TE YANG Module

The RSVP-TE generic YANG module "ietf-rsvp-te" imports the following modules:

- ietf-rsvp defined in [I-D.ietf-teas-yang-rsvp]
- ietf-routing-types defined in [RFC8294]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]
- ietf-te and ietf-te-dev defined in [I-D.ietf-teas-yang-te]

This module references the following documents:
[I-D.ietf-teas-yang-rsvp], [RFC8349], [I-D.ietf-teas-yang-te],
[I-D.ietf-teas-yang-te-types], [RFC2210], [RFC4920], [RFC5420],
[RFC7570], [RFC4859].

```yml
<CODE BEGINS> file "ietf-rsvp-te@2019-04-09.yang"
module ietf-rsvp-te {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp-te";
  prefix "rsvp-te";

  import ietf-rsvp {
    prefix rsvp;
    reference "draft-ietf-teas-yang-rsvp: A YANG Data Model for
    Resource Reservation Protocol (RSVP)";
  }

  import ietf-routing {
    prefix "rt";
    reference "RFC8349: A YANG Data Model for Routing Management";
  }

  import ietf-routing-types {
    prefix rt-types;
    reference "RFC8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-te {
    prefix te;
```
reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

import ietf-te-device {
  prefix te-dev;
  reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

/* Import TE generic types */
import ietf-te-types {
  prefix te-types;
  reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering Types";
}

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

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

WG Chair: Lou Berger
<mailto:lberger@labn.net>

WG Chair: Vishnu Pavan Beeram
<mailto:vbeeram@juniper.net>

Editor: Vishnu Pavan Beeram
<mailto:vbeeram@juniper.net>

Editor: Tarek Saad
<mailto:tsaad.net@gmail.com>

Editor: Rakesh Gandhi
<mailto:rgandhi@cisco.com>

Editor: Xufeng Liu
<mailto: xufeng.liu.ietf@gmail.com>

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

Editor: Himanshu Shah
<mailto:hshah@ciena.com>";
description
"This module contains the RSVP-TE YANG generic data model.
The model fully conforms to the Network Management Datastore Architecture (NMDA).

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (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."

// RFC Ed.: replace XXXX with actual RFC number and remove this
// note.

// RFC Ed.: update the date below with the date of RFC publication
// and remove this note.

revision "2019-04-09" {
  description "A YANG Data Model for RSVP-TE";
  reference "RFCXXXX: A YANG Data Model for RSVP-TE Protocol";
}

identity rsvp-message-type {
  description "RSVP message types";
}

identity rsvp-message-path {
  base rsvp-message-type;
  description "RSVP Path message";
  reference "RFC2205";
}

identity rsvp-message-resv {
  base rsvp-message-type;
  description "RSVP Resv message";
  reference "RFC2205";
}

identity rsvp-message-path-err {
  base rsvp-message-type;
  description "RSVP Path-Err message";
  reference "RFC2205";
}
identity rsvp-message-resv-err {
    base rsvp-message-type;
    description "RSVP Resv-Err message";
    reference "RFC2205";
}

identity rsvp-message-path-tear {
    base rsvp-message-type;
    description "RSVP Path Tear message";
    reference "RFC2205";
}

identity rsvp-message-resv-conf {
    base rsvp-message-type;
    description "RSVP Resv Confirm message";
    reference "RFC2205";
}

identity rsvp-message-srefresh {
    base rsvp-message-type;
    description "RSVP SRefresh message";
    reference "RFC2961";
}

identity rsvp-message-hello {
    base rsvp-message-type;
    description "RSVP Hello message";
    reference "RFC3209";
}

identity rsvp-message-bundle {
    base rsvp-message-type;
    description "RSVP Bundle message";
    reference "RFC2961";
}

identity rsvp-message-notify {
    base rsvp-message-type;
    description "RSVP Notify message";
    reference "RFC3473";
}

/*
 * RSVP-TE LSPs groupings.
 */
grouping lsp-record-route-information-state {
    description "recorded route information grouping";
    container incoming-record-route-subobjects {
        description "RSVP recorded route object incoming information";
        list incoming-record-route-subobject {
            when "../../te:origin-type != 'ingress'" {
                description "Applicable on non-ingress LSPs only";
            }
            key "index";
        }
    }
}
container outgoing-record-route-subobjects {
  description "RSVP recorded route object outgoing information";
  list outgoing-record-route-subobject {
    when "../../te:origin-type != 'egress'" {
      description "Applicable on non-egress LSPs only";
    }
    key "index";
    ordered-by user;
    description "List of RSVP Resv record-route objects";
    uses te-types:record-route-state;
  }
}

grouping lsp-explicit-route-information-state {
  description "RSVP-TE LSP explicit-route information";
  container explicit-route-objects {
    description "Explicit route object information";
    list incoming-explicit-route-hop {
      when "../../te:origin-type != 'ingress'" {
        description "Applicable on non-ingress LSPs only";
      }
      key "index";
      ordered-by user;
      description "List of incoming RSVP Path explicit-route objects";
      leaf index {
        type uint32;
        description "Explicit route hop index. The index is used to
        identify an entry in the list. The order of entries
        is defined by the user without relying on key values";
      }
      uses te-types:explicit-route-hop;
    }
    list outgoing-explicit-route-hop {
      when "../../te:origin-type != 'egress'" {
        description "Applicable on non-egress LSPs only";
      }
      key "index";
      ordered-by user;
    }
  }
}
description
"List of outgoing RSVP Path explicit-route objects";
leaf index {
  type uint32;
description
  "Explicit route hop index. The index is used to identify an entry in the list. The order of entries is defined by the user without relying on key values";
}
uses te-types:explicit-route-hop;
}
}

grouping lsp-attributes-flags-config {
  description
  "Configuration parameters relating to RSVP-TE LSP attribute flags";
  leaf-list lsp-attribute {
    type identityref {
      base te-types:lsp-attributes-flags;
    }
description "RSVP per LSP attributes flags";
reference
  "RFC4920, RFC5420, RFC7570";
}
}

grouping lsp-session-attributes-obj-flags-config {
  description
  "Configuration parameters relating to RSVP-TE LSP session attribute flags";
reference
  "RFC4859: Registry for RSVP-TE Session Flags";
  leaf-list session-attribute {
    when ".../session-attribute != ‘te-types:bandwidth-protection-desired’ or
    .../session-attribute != ‘te-types:soft-preemption-desired’"
    type identityref {
      base te-types:session-attributes-flags;
    }
description "RSVP session attributes flags";
reference
  "RFC4859: Registry for RSVP-TE Session Flags";
}
}
grouping lsp-properties-config {
  description "Configuration parameters relating to RSVP-TE LSP session attribute flags";
  leaf lsp-signaled-name {
    type string;
    description "Sets the session name to use in the session attribute object.";
  }
  uses lsp-session-attributes-obj-flags-config;
  uses lsp-attributes-flags-config;
}

grouping tunnel-properties-config {
  description "RSVP-TE Tunnel properties grouping";
  leaf retry-timer {
    type uint16 {
      range 1..600;
    }
    units seconds;
    description "sets the time between attempts to establish the LSP";
  }
}

/*** End of RSVP-TE LSP groupings ***/

/**
 * RSVP-TE generic global properties.
 */
grouping global-soft-preemption-config {
  description "Configuration for global RSVP-TE soft preemption";
  leaf soft-preemption-timeout {
    type uint16 {
      range 0..300;
    }
    default 0;
    description "Timeout value for soft preemption to revert to hard preemption";
  }
}

grouping global-soft-preemption {
  description
"Top level group for RSVP-TE soft-preemption";
container global-soft-preemption {
    presence "Enables soft preemption on a node.";
    description
        "Top level container for RSVP-TE soft-preemption";
    uses global-soft-preemption-config;
}
} /* End of RSVP-TE generic global properties. */

/**
 * RSVP-TE interface generic groupings.
 */
grouping rsvp-te-interface-attributes {
    description
        "Top level grouping for RSVP-TE interface properties.";
    container rsvp-te-interface-attributes {
        description
            "Top level container for RSVP-TE interface properties";
        container state {
            config false;
            description
                "State information associated with RSVP-TE bandwidth";
        }
    }
} /* End of RSVP-TE generic groupings */

/* RSVP-TE global properties */
    description
        "RSVP-TE augmentation to RSVP globals";
    uses global-soft-preemption;
}

/* Linkage to the base RSVP all links */
    description
        "RSVP-TE generic data augmentation pertaining to interfaces";
    uses rsvp-te-interface-attributes;
}

/* Linkage to per RSVP interface */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp/rsvp/rsvp:interfaces/" +
  "rsvp:interface" {
    description
    "RSVP-TE generic data augmentation pertaining to specific
    interface";
    uses rsvp-te-interface-attributes;
  }
/* add augmentation for sessions and neighbors */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp/rsvp/rsvp:globals/
  + rsvp:sessions/rsvp:session/rsvp:state/rsvp:psbs/rsvp:psb" {
    description
    "RSVP-TE generic data augmentation pertaining to session";
    /* To be added */
    leaf tspec-average-rate {
      type rt-types:bandwidth-ieee-float32;
      units "Bytes per second";
      description "Tspec Token Bucket Average Rate";
      reference "RFC2210: RSVP with INTSERV";
    }
    leaf tspec-size {
      type rt-types:bandwidth-ieee-float32;
      units "Bytes per second";
      description "Tspec Token Bucket Burst Rate";
      reference "RFC2210";
    }
    leaf tspec-peak-rate {
      type rt-types:bandwidth-ieee-float32;
      units "Bytes per second";
      description "Tspec Token Bucket Peak Data Rate";
      reference "RFC2210";
    }
    leaf min-policed-unit {
      type uint32;
      description "Tspec Minimum Policed Unit";
      reference "RFC2210";
    }
    leaf max-packet-size {
      type uint32;
      description "Tspec Maximum Packet Size";
      reference "RFC2210";
    }
  }
/* add augmentation for sessions and neighbors */
description
  "RSVP-TE generic data augmentation pertaining to session";
leaf fspec-average-rate {
  type rt-types:bandwidth-ieee-float32;
  units "Bytes per second";
  description "Fspec Token Bucket Average Rate";
  reference "RFC2210";
}
leaf fspec-size {
  type rt-types:bandwidth-ieee-float32;
  units "Bytes per second";
  description "Fspec Token Bucket Burst Rate";
  reference "RFC2210";
}
leaf fspec-peak-rate {
  type rt-types:bandwidth-ieee-float32;
  units "Bytes per second";
  description "Fspec Token Bucket Peak Data Rate";
  reference "RFC2210";
}
leaf min-policed-unit {
  type uint32;
  description "Fspec Minimum Policed Unit";
  reference "RFC2210";
}
leaf max-packet-size {
  type uint32;
  description "Fspec Maximum Packet Size";
  reference "RFC2210";
}
}
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:neighbors" {
  description
  "RSVP-TE generic data augmentation pertaining to neighbors";
  /* To be added */
}
/**
 * RSVP-TE generic augmentations of generic TE model.
 */
/*
 * TE tunnel augmentation */
augment "/te:te/te:tunnels/te:tunnel" {
  when "/te:te/te:tunnels/te:tunnel" +
    "/te:p2p-primary-paths/te:p2p-primary-path" +
    "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" {
description
  "When the path signaling protocol is RSVP-TE ";
}
description
  "RSVP-TE generic data augmentation pertaining to TE tunnels";
uses lsp-properties-config;
uses tunnel-properties-config;
}

/* TE LSP augmentation */
grouping rsvp-te-lsp-error-info {
  description
    "Grouping for RSVP-TE error reporting information";
  leaf rsvp-message-type {
    type identityref {
      base rsvp-message-type;
    }
  }
  description
    "The RSVP message type that delivered the error";
}
leaf rsvp-error-code {
  type uint8;
  description "RSVP error code";
  reference "RFC2205";
}
leaf rsvp-error-subcode {
  type uint16;
  description "RSVP Error sub-codes";
  reference "RFC2205";
}
}

grouping rsvp-te-lsp-properties {
  description "RSVP-TE LSP properties grouping";
  leaf associated-rsvp-session {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/
        + "rt:control-plane-protocol/rsvp:rsvp/globals/
        + "rsvp:sessions/rsvp:session/rsvp:local-index";
    }
    config false;
    description
      "If the signalling protocol specified for this path is RSVP-TE, this leaf provides a reference to the associated session within the RSVP-TE protocol sessions list, such that details of the signaling can be retrieved.";
  }
}

uses lsp-properties-config;
uses lsp-explicit-route-information-state;
uses lsp-record-route-information-state;
}

augment "/te:te/te:lsps-state/te:lsp" {
  when "/te:te/te:lsps-state/te:lsp" + 
    "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" { 
    description
    "When the signaling protocol is RSVP-TE ";
  } 
  description
  "RSVP-TE generic data augmentation pertaining to specific TE LSP";
  uses rsvp-te-lsp-properties;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
  "/te:p2p-primary-path/te:lsps/te:lsp" {
  when "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
    "/te:p2p-primary-path/te:lsps/te:lsp" + 
    "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" { 
    description
    "When the signaling protocol is RSVP-TE ";
  } 
  description
  "RSVP-TE generic data augmentation pertaining to specific TE LSP";
  uses rsvp-te-lsp-properties;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
  "/te:p2p-primary-path/te:lsps/te:lsp/provisioning-error-infos" + 
  "/te:lsp-provisioning-error-info" { 
  description
  "Augmentation for RSVP-TE per LSP error reason"; 
  uses rsvp-te-lsp-error-info;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
  "/te:p2p-primary-path/te:lsps/te:lsp/provisioning-error-infos" + 
  "/te:lsp-provisioning-error-info" {
  when "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
    "/te:p2p-primary-path/te:lsps/te:lsp" + 
    "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" { 
    description
    "When the signaling protocol is RSVP-TE ";
  } 
}
augment "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
    "/te:p2p-secondary-path/te:lsps/te:lsp" { 
    when "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
        "/te:p2p-secondary-path/te:lsps/te:lsp" + 
        "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" { 
        description 
        "When the signaling protocol is RSVP-TE "; 
    } 
} 

description 
"RSVP-TE generic data augmentation pertaining to specific TE 
LSP"; 
uses rsvp-te-lsp-properties; 
} 

description 
"RSVP-TE generic data augmentation pertaining to specific TE 
interface"; 
} 

/* TE interface augmentation */ 
augment "/te:te-dev:interfaces/te-dev:interface" { 
    description 
    "RSVP-TE generic data augmentation pertaining to specific TE 
    interface"; 
} 
}
2.3.2. RSVP-TE MPLS YANG Module

The RSVP-TE MPLS YANG module "ietf-rsvp-te-mpls" imports the following module(s):

- ietf-rsvp defined in [I-D.ietf-teas-yang-rsvp]
- ietf-routing-types defined in [RFC8294]
- ietf-te-mpls-types defined in [I-D.ietf-teas-yang-te-types]
- ietf-te and ietf-te-dev defined in [I-D.ietf-teas-yang-te]

This module references the following documents:
[I-D.ietf-teas-yang-rsvp], [RFC8349], [I-D.ietf-teas-yang-te-types],
[I-D.ietf-teas-yang-te], [RFC3209].
reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering Types";
}

import ietf-te {
  prefix "te";
  reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

import ietf-te-device {
  prefix "te-dev";
  reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

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

contact
  "WG Web:  <http://tools.ietf.org/wg/teas/>
  WG List:  <mailto:teas@ietf.org>
  WG Chair: Lou Berger
            <mailto:lberger@labn.net>
  WG Chair: Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
  Editor:  Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
  Editor:  Tarek Saad
            <mailto:tsaad.net@gmail.com>
  Editor:  Rakesh Gandhi
            <mailto:rgandhi@cisco.com>
  Editor:  Xufeng Liu
            <mailto: xufeng.liu.ietf@gmail.com>
  Editor:  Igor Bryskin
            <mailto:Igor.Bryskin@huawei.com>
  Editor:  Himanshu Shah
            <mailto:hshah@ciena.com">";
description
"Latest update to MPLS RSVP-TE YANG data model. The model fully conforms to the Network Management Datastore Architecture (NMDA).

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (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."

// RFC Ed.: replace XXXX with actual RFC number and remove this note.

// RFC Ed.: update the date below with the date of RFC publication // and remove this note.

revision "2019-04-09" {
  description "Update to MPLS RSVP-TE YANG initial revision.";
  reference
    "RFCXXXX: A YANG Data Model for RSVP-TE Protocol";
}

/* RSVP-TE MPLS LSPs groupings */
grouping lsp-attributes-flags-mpls-config {
  description
    "Configuration parameters relating to RSVP-TE MPLS LSP attribute flags";
}

grouping lsp-session-attributes-obj-flags-mpls-config {
  description
    "Configuration parameters relating to RSVP-TE MPLS LSP session attribute flags";
  reference
    "RFC4859: Registry for RSVP-TE Session Flags";
  leaf-list session-attribute {
    when "../session-attribute = 'te-types:bandwidth-protection-desired' or
    ../session-attribute = 'te-types:soft-preemption-desired'";
    type identityref {

grouping tunnel-properties-mpls-config {
  description "Top level grouping for LSP properties.";
  uses lsp-session-attributes-obj-flags-mpls-config;
  uses lsp-attributes-flags-mpls-config;
}

/* End of RSVP-TE MPLS LSPs groupings */

/* MPLS RSVP-TE interface groupings */
grouping rsvp-te-interface-state {
  description "The RSVP-TE interface state grouping";
  leaf over-subscribed-bandwidth {
    type te-packet-types:bandwidth-kbps;
    description "The amount of over-subscribed bandwidth on
    the interface";
  }
}

grouping rsvp-te-interface-softpreemption-state {
  description "The RSVP-TE interface preemptions state grouping";
  container interface-softpreemption-state {
    description "The RSVP-TE interface preemptions state grouping";
    leaf soft-preempted-bandwidth {
      type te-packet-types:bandwidth-kbps;
      description "The amount of soft-preempted bandwidth on
      this interface";
    }
    list lsps {
      ...
key
"source destination tunnel-id lsp-id "+
"extended-tunnel-id";
description
"List of LSPs that are soft-preempted";
leaf source {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/"
    "te:source";
  }
description
  "Tunnel sender address extracted from
  SENDER_TEMPLATE object";
  reference "RFC3209";
}
leaf destination {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/"
    "te:destination";
  }
description
  "Tunnel endpoint address extracted from
  SESSION object";
  reference "RFC3209";
}
leaf tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/"
    "te:tunnel-id";
  }
description
  "Tunnel identifier used in the SESSION
  that remains constant over the life
  of the tunnel.";
  reference "RFC3209";
}
leaf lsp-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/"
    "te:lsp-id";
  }
description
  "Identifier used in the SENDER TEMPLATE
  and the FILTER_SPEC that can be changed
to allow a sender to share resources with
  itself.";
  reference "RFC3209";
leaf extended-tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/" +
    "te:extended-tunnel-id";
  }
  description
  "Extended Tunnel ID of the LSP.";
  reference "RFC3209";
}
leaf type {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/" +
    "te:type";
  }
  description "LSP type P2P or P2MP";
}
}
}

grouping bandwidth-mpls-constraints {
  description "Bandwidth constraints.";
  container bandwidth-mpls-constraints {
    description
    "Holds the bandwidth constraints properties";
    leaf maximum-reservable {
      type te-packet-types:bandwidth-kbps;
      description
      "The maximum reservable bandwidth on the
      interface in kbps";
    }
    leaf-list bc-value {
      type uint32 {
        range "0..4294967295";
      }
      max-elements 8;
      description
      "The bandwidth constraint type";
    }
  }
}
}

grouping bandwidth-constraint-values {
  description
  "Packet bandwidth contraints values";
  choice value-type {
    description
    "Value representation";
  }
}
case percentages {
    container perc-values {
        uses bandwidth-mpls-constraints;
        description "Percentage values";
    }
}
case absolutes {
    container abs-values {
        uses bandwidth-mpls-constraints;
        description "Absolute values";
    }
}
}
grouping bandwidth-mpls-reservable-config {
    description "Interface bandwidth reservable configuration grouping";
    choice bandwidth-value {
        description "Reservable bandwidth configuration choice";
        case absolute {
            leaf absolute-value {
                type te-packet-types:bandwidth-kbps;
                description "Absolute value of the bandwidth";
            }
        }
        case percentage {
            leaf percent-value {
                type uint32 {
                    range "0..4294967295";
                }
                description "Percentage reservable bandwidth";
            }
            description "The maximum reservable bandwidth on the interface";
        }
    }
    choice bc-model-type {
        description "Reservable bandwidth percentage capacity values.";
        case bc-model-rdm {
            container bc-model-rdm {
                description "Russian Doll Model Bandwidth Constraints.";
            }
        }
    }
}
uses bandwidth-mpls-constraints;
}
}
case bc-model-mam {
  container bc-model-mam {
    uses bandwidth-mpls-constraints;
    description
    "Maximum Allocation Model Bandwidth Constraints.";
  }
}
case bc-model-mar {
  container bc-model-mar {
    uses bandwidth-mpls-constraints;
    description
    "Maximum Allocation with Reservation Model Bandwidth Constraints.";
  }
}
}
}
}

grouping bandwidth-mpls-reservable {
  description
  "Packet reservable bandwidth";
  container bandwidth-mpls-reservable {
    description
    "Interface bandwidth reservable container";
    uses bandwidth-mpls-reservable-config;
  }
}

/*@ End of RSVP-TE interface groupings */

/*@ RSVP-TE FRR groupings */
grouping rsvp-te-frr-auto-tunnel-backup-config {
  description
  "Auto-tunnel backup configuration grouping";
  leaf auto-backup-protection {
    type identityref {
      base te-packet-types:backup-protection-type;
    }
    default
    te-packet-types:backup-protection-node-link;
    description
    "Describes whether the backup should offer protection against link, node, or either";
  }
  leaf auto-backup-path-computation {
type identityref {
    base
    te-types:path-computation-srlg-type;
}

description
    "FRR backup computation type";
}

grouping rsvp-te-frr-backups-config {
    description
        "Top level container for RSVP-TE FRR backup parameters";
    choice type {
        description
            "FRR backup tunnel type";
        case static-tunnel {
            container static-backups {
                description
                    "List of static backups";
                list static-backup {
                    key "backup-tunnel-name";
                    description
                        "List of static backup tunnels that protect the RSVP-TE interface.";
                    leaf backup-tunnel-name {
                        type leafref {
                            path "/te:te/te:tunnels/te:tunnel/te:name";
                        }
                        description
                            "FRR Backup tunnel name";
                    }
                }
            }
            
            case auto-tunnel {
                container auto-tunnel-backups {
                    description
                        "Auto-tunnel choice";
                    uses rsvp-te-frr-auto-tunnel-backup-config; 
                }
            }
            
        }
    }
}


grouping rsvp-te-frr-backups {
    description
        "RSVP-TE facility backup grouping";
    container rsvp-te-frr-backups {
        description
            "RSVP-TE facility backup properties";
        uses rsvp-te-frr-backups-config; 
    }
}
grouping lsp-backup-info-state {
   description "LSP backup information grouping";
   leaf backup-tunnel-name {
      type string;
      description
      "If an LSP has an FRR backup LSP that can protect it, this field identifies the tunnel name of the backup LSP. Otherwise, this field is empty.";
   } leaf backup-frr-on {
      type uint8;
      description
      "Whether currently this backup is carrying traffic";
   } leaf backup-protected-lsp-num {
      type uint32;
      description
      "Number of LSPs protected by this backup";
   } }
}

grouping lsp-backup-info {
   description "Backup/bypass LSP related information";
   container backup-info {
      description
      "backup information";
      uses lsp-backup-info-state;
   } }
}

grouping fast-reroute-local-revertive-config {
   description "RSVP-TE FRR local revertive grouping";
   leaf rsvp-frr-local-revert-delay {
      type uint32;
      description
      "Time to wait after primary link is restored before node attempts local revertive procedures.";
   } }
}

/*** End of RSVP-TE FRR backup information ***/

grouping fast-reroute-local-revertive {
   description
   "RSVP-TE FRR local revertive grouping";
   leaf rsvp-frr-local-revert-delay {
      type uint32;
      description
      "Time to wait after primary link is restored before node attempts local revertive procedures.";
   } }

/*** End of RSVP-TE FRR backup information ***/
"Top level grouping for globals properties";
container fast-reroute-local-revertive {
    description "RSVP-TE FRR local revertive container";
    uses fast-reroute-local-revertive-config;
}
/* RSVP-TE global properties */
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp" { 
    description 
    "RSVP-TE augmentation to RSVP globals";
    uses fast-reroute-local-revertive;
}
/* Linkage to the base RSVP all interfaces */
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces" { 
    description 
    "Augmentations for RSVP-TE MPLS all interfaces properties";
    /* To be added */
}
/* Linkage to per RSVP interface */
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/" + 
"rsvp:interface" { 
    description 
    "Augmentations for RSVP-TE MPLS per interface properties";
    /* To be added */
}
/* add augmentation for sessions neighbors */
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" 
+ "rsvp:sessions/rsvp:session/rsvp:state" { 
    description 
    "Augmentations for RSVP-TE MPLS sessions";
    /* To be added */
}
augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:neighbors" { 
    description 
    "Augmentations for RSVP-TE MPLS neighbors properties";
    /* To be added */
}
/**
 * Augmentation to TE generic module
 */

augment "/te:te:te:tunnels/te:tunnel"
{
  description
  "Augmentations for RSVP-TE MPLS TE tunnel properties"
  uses tunnel-properties-mpls-config;
}

augment "/te:te:te:lsps-state/te:lsp"
{
  when "/te:te:te:lsps-state/te:lsp" +
  "/te:path-setup-protocol = 'te-types:path-setup-rsvp'"
  {
    description
    "When the signaling protocol is RSVP-TE ";
  }
  description
  "RSVP-TE MPLS LSP state properties"
  uses lsp-properties-mpls;
  uses lsp-backup-info;
}

augment "/te:te:te:tunnels/te:tunnel/te:p2p-primary-paths" +
  "/te:p2p-primary-path/te:lsps/te:lsp"
{
  when "/te:te:te:tunnels/te:tunnel" +
  "/te:p2p-secondary-paths/te:p2p-secondary-path/" +
  "te:path-setup-protocol = 'te-types:path-setup-rsvp'"
  {
    description
    "When the signaling protocol is RSVP-TE ";
  }
  description
  "RSVP-TE MPLS LSP state properties"
  uses lsp-properties-mpls;
  uses lsp-backup-info;
}

augment "/te:te:te:tunnels/te:tunnel/te:p2p-secondary-paths" +
  "/te:p2p-secondary-path/te:lsps/te:lsp"
{
  when "/te:te:te:tunnels/te:tunnel" +
  "/te:p2p-secondary-paths/te:p2p-secondary-path/" +
  "te:path-setup-protocol = 'te-types:path-setup-rsvp'"
  {
    description
    "When the signaling protocol is RSVP-TE ";
  }
  description
  "RSVP-TE MPLS LSP state properties"
  uses lsp-properties-mpls;
  uses lsp-backup-info;
}
augment "/te:te/te-dev:interfaces/te-dev:interface" {
    description
        "RSVP reservable bandwidth configuration properties";
    uses bandwidth-mpls-reservable;
}

augment "/te:te/te-dev:interfaces/te-dev:interface" {
    description
        "RSVP reservable bandwidth configuration properties";
    uses rsvp-te-frr-backups;
}
}

<CODE ENDS>

Figure 5: RSVP TE MPLS YANG module

3. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

XML: N/A, the requested URI is an XML namespace.

XML: N/A, the requested URI is an XML namespace.

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

name:       ietf-rsvp
prefix:     ietf-rsvp
reference:  RFCXXXX

name:       ietf-rsvp-te
prefix:     ietf-rsvp-te
reference:  RFCXXXX

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF
users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module(s) defined in this document 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., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp/globals: The data nodes defined in this document and under this branch are applicable device-wide and can affect all RSVP established sessions. Unauthorized access to this container can potentially cause disruptive event(s) on all established sessions.

/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/rsvp:sessions: The data nodes defined in this document and under this branch are applicable to one or all RSVP-TE session(s). Unauthorized access to this container can potentially affect the impacted RSVP session(s).

/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces: The data nodes defined in this document and under this branch are applicable to one or all RSVP interfaces. Unauthorized access to this container can potentially affect established session(s) over impacted interface(s).

5. Acknowledgement

The authors would like to thank Lou Berger for reviewing and providing valuable feedback on this document.

6. Contributors
7. References

7.1. Normative References

[I-D.ietf-teas-yang-rsvp]

[I-D.ietf-teas-yang-te]

[I-D.ietf-teas-yang-te-types]


7.2. Informative References


Authors' Addresses

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net
Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems, Inc.
Email: rgandhi@cisco.com

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

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

Himanshu Shah
Ciena
Email: hshah@ciena.com
YANG Data Model for SR and SR TE Topologies
draft-ietf-teas-yang-sr-te-topo-04

Abstract

This document defines a YANG data model for Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology.

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 September 12, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of
draft-ietf-teas-yang-sr-te-topo-04.
publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction .............................................. 2
   1.1. Terminology ........................................ 2
   1.2. Tree Diagrams ...................................... 3
2. Modeling Considerations ................................. 3
   2.1. Segment Routing (SR) Topology ....................... 3
   2.2. Segment Routing (SR) TE Topology ................... 3
   2.3. Relations to ietf-segment-routing ................... 4
   2.4. Topology Type Modeling .............................. 5
   2.5. Topology Attributes ................................ 5
   2.6. Node Attributes ..................................... 5
   2.7. Link Attributes ..................................... 6
3. Model Structure .......................................... 7
4. YANG Module ............................................. 9
5. IANA Considerations ..................................... 16
6. Security Considerations ................................. 17
7. References ............................................... 18
   7.1. Normative References ............................... 18
   7.2. Informative References ............................. 19
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations ........................................... 21
   A.1. SR Topology State Module ............................. 21
Appendix B. Data Tree Example .............................. 24
Appendix C. Contributors ................................... 31
Authors’ Addresses ......................................... 31

1. Introduction

This document defines a YANG [RFC7950] data model for describing the presentations of Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology. The version of the model limits the transport type to an MPLS dataplane.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.
The following terms are defined in [RFC7950] and are not redefined here:

- augment
- data model
- data node

1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Modeling Considerations

2.1. Segment Routing (SR) Topology

The Layer 3 network topology model is discussed in [RFC8346]. The Segment Routing (SR) topology model proposed in this document augments and uses the ietf-l3-unicast-igp-topology module defined in [RFC8346]. SR related attributes are covered in the ietf-sr-topology model.

```
+------------------------+
|       SR Topology      |
|     ietf-sr-topology   |
+------------------------+
    V
+------------------------+
|   SR Topology          |
|   ietf-sr-topology     |
+------------------------+
     +------------------------+
     |         Layer 3 Network Topology |
     |     ietf-l3-unicast-topology   |
     +------------------------+
```

2.2. Segment Routing (SR) TE Topology

When traffic engineering is enabled on an SR topology, there will be associations between objects in SR topologies and objects in TE topologies. An SR TE topology is both an SR topology and a layer 3 TE topology. Multiple inheritance is used to achieve such relations.
Each type of topologies is indicated by "network-types" defined in [RFC8345]. For the three types of topologies above, the data representations are:

L3 Topology:
/nd:networks/nd:network/nd:network-types/l3-unicast-topology

L3 TE Topology:
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te

SR Topology:

SR TE Topology: (multiple inheritance)

2.3. Relations to ietf-segment-routing

[I-D.ietf-spring-sr-yang] defines ietf-segment-routing that is a model intended to be used on network elements to configure or operate segment routing; ietf-sr-topology defined in this document is intended to be used on a controller for the network-wide operations such as path computation.

SR topology model shares many modeling constructs defined in ietf-segment-routing. The module ietf-sr-topology uses the types and groupings defined in ietf-segment-routing.
2.4. Topology Type Modeling

A new topology type is defined in this document, to indicate a topology that is a Segment Routing (SR) topology on an MPLS dataplane.

```
augment /nw:networks/nw:network/nw:network-types
  /l3t:l3-unicast-topology:
    +--rw sr-mpls!
```

2.5. Topology Attributes

The Segment Routing attributes with topology-wide impacts are modeled by augmenting the container "l3-topology-attributes" in the L3 topology model. SRGB (Segment Routing Global Block) is covered in this augmentation. A SR domain is mapped to a topology in this model.

```
augment /nw:networks/nw:network/l3t:l3-topology-attributes:
  +--rw sr
    +--rw srgb* [lower-bound upper-bound]
    +--rw lower-bound    uint32
    +--rw upper-bound    uint32
```

2.6. Node Attributes

The Segment Routing attributes within the node scope are modeled by augmenting the sub tree /nw:networks/nw:network/nw:node/ in the L3 topology model.

The SR attributes that have node-scope impact are modeled by augmenting the container "l3-node-attributes" in the L3 topology model, including the SR capabilities, SRGB (Segment Routing Global Block), and SRLB (Segment Routing Local Block) specified on this mode. This model also provides the information about how these SR attributes are learned:
augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
  +++-rw sr
  |      +++-rw srgb* [lower-bound upper-bound]
  |            |      |      +++-rw lower-bound  uint32
  |            |      |      +++-rw upper-bound  uint32
  |      +++-rw srlb* [lower-bound upper-bound]
  |            |      +++-rw lower-bound  uint32
  |            |      +++-rw upper-bound  uint32
  +++-ro node-capabilities
  |      +++-ro transport-planes* [transport-plane]
  |            |      +--ro transport-plane  identityref
  |      +++-ro entropy-readable-label-depth?  uint8
  +++-rw msd?  uint8 {msd}?
  +++-ro information-source?  enumeration
  +++-ro information-source-state
  +++-ro credibility-preference?  uint16

The SR attributes that are related to a IGP-Prefix segment are modeled by augmenting the list entry "prefix" in the L3 topology model:

augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes
  /l3t:prefix:
  +++-rw sr!
  |      +++-rw value-type?  enumeration
  |      +++-rw start-sid  uint32
  |      +++-rw range?  uint32
  |      +++-rw algorithm?  identityref
  |      +++-rw last-hop-behavior?  enumeration
  |          |      (sid-last-hop-behavior)?
  |      +++-rw is-local?  boolean
  |      +++-rw is-node?  boolean
  |      +++-ro is-readvertisment?  boolean

2.7. Link Attributes

A link in the topology model connects the termination point on the source node to the termination point on the destination node. When such a link is instantiated, the bindings between the nodes and the corresponding Adj-SIDs are formed, and the resulting FIB entries are installed.

A link in the topology model is mapped to an SR Adjacency Segment, formed by a pair of interfaces on two respective adjacent nodes. The SR Adjacency Segment attributes are modeled by augmenting the link attributes of the L3 topology model. The modeling structure is as follows:
module: ietf-sr-topology
    augment /nw:networks/nw:network/nw:node/l3t:13-node-attributes:
        +++-rw sr
            +++-rw srgb* [lower-bound upper-bound]
                |  +++-rw lower-bound   uint32
                |  +++-rw upper-bound   uint32

| +--rw upper-bound    uint32
| +--rw lower-bound    uint32
| +--rw upper-bound    uint32
| +--ro node-capabilities
| | +--ro transport-planes* [transport-plane]
| | | +--ro transport-plane    identityref
| | +--ro entropy-readable-label-depth?   uint8
| +--rw msd?                           uint8 {msd}?
| +--ro information-source?   enumeration
| +--ro information-source-instance?   string
| +--ro information-source-state
| | +--ro credibility-preference?   uint16
augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes
| /l3t:prefix:
| +--rw sr!
| | +--rw value-type?          enumeration
| | +--rw start-sid            uint32
| | +--rw range?               uint32
| | +--rw algorithm?           identityref
| | +--rw last-hop-behavior?   enumeration
| | | {sid-last-hop-behavior}?
| | +--rw is-local?            boolean
| | +--rw is-node?             boolean
| | +--ro is-readvertisement?  boolean
augment /nw:networks/nw:network/nt:link/l3t:l3-link-attributes:
| +--rw sr!
| | +--rw value-type?          enumeration
| | +--rw sid                  uint32
| | +--rw advertise-protection? enumeration
| | +--rw is-local?            boolean
| | +--rw msd?                 uint8 {msd}?
| | +--rw address-family?      enumeration
| | +--rw is-backup?           boolean
| | +--rw is-part-of-set?      boolean
| | +--rw is-persistent?       boolean
| | +--rw is-on-lan?           boolean
| | +--ro information-source?  enumeration
| | +--ro information-source-instance?   string
| | +--ro information-source-state
| | | +--ro credibility-preference?   uint16
4. YANG Module

<CODE BEGINS> file "ietf-sr-topology@2019-03-09.yang"
module ietf-sr-topology {
    yang-version 1.1;
    prefix "srt";

    import ietf-network {
        prefix "nw";
        reference "RFC 8345: A YANG Data Model for Network Topologies";
    }
    import ietf-network-topology {
        prefix "nt";
        reference "RFC 8345: A YANG Data Model for Network Topologies";
    }
    import ietf-l3-unicast-topology {
        prefix "l3t";
        reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
    }
    import ietf-segment-routing-common {
        prefix "sr-cmn";
        reference "I-D.ietf-spring-sr-yang: YANG Data Model for Segment Routing";
    }

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

    contact
        "WG Web:   <http://tools.ietf.org/wg/teas/>
        WG List:  <mailto:teas@ietf.org>
        Editor:   Xufeng Liu
                  <mailto:xufeng.liu.ietf@gmail.com>
        Editor:   Igor Bryskin
                  <mailto:Igor.Bryskin@huawei.com>
        Editor:   Vishnu Pavan Beeram
                  <mailto:vbeeram@juniper.net>
        Editor:   Tarek Saad
                  <mailto:tsaad@cisco.com>
        Editor:   Himanshu Shah

Internet-Draft             YANG SR TE Topology                March 2019

Editor: Stephane Litkowski
<mailto:stephane.litkowski@orange.com>

description
"YANG data model for representing and manipulating Segment Routing Topologies.

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

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

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

revision 2019-03-09 {
  description "Initial revision";
  reference
    "RFC XXXX: YANG Data Model for SR and SR TE Topologies";
}

feature msd {
  description
    "Support of signaling MSD (Maximum SID Depth) in IGP.";
}

grouping sr-topology-type {
  description
    "Identifies the SR-MPLS topology type. This type of network topologies use Segment Routing (SR) technology over the MPLS data plane";
  container sr-mpls {
    presence "Indicates SR-MPLS topology";
    description
      "Its presence identifies the SR topology type.";
  }
}

augment "/nw:networks/nw:network/nw:network-types/
  + "l3t:l3-unicast-topology" {
    description

"Defines the SR topology type.";
uses sr-topology-type;
}
anugment "/nw:networks/nw:network/l3t:l3-topology-attributes" {
when "../nw:network-types/l3t:l3-unicast-topology/srt:sr-mpls" {
    description "Augment only for SR topology.";
}
description "Augment topology configuration";
uses sr-topology-attributes;
}
anugment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes" {
when "../../nw:network-types/l3t:l3-unicast-topology/"
+ "srt:sr-mpls" {
    description "Augment only for SR topology.";
}
description "Augment node configuration.";
uses sr-node-attributes;
}
anugment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes"
+ "/l3t:prefix" {
when "../../nw:network-types/l3t:l3-unicast-topology/"
+ "srt:sr-mpls" {
    description "Augment only for SR topology.";
}
description "Augment node prefix.";
uses sr-node-prefix-attributes;
}
anugment "/nw:networks/nw:network/nt:link/l3t:l3-link-attributes" {
when "../../nw:network-types/l3t:l3-unicast-topology/"
+ "srt:sr-mpls" {
    description "Augment only for SR topology.";
}
description "Augment link configuration";
uses sr-link-attributes;
}
grouping sr-topology-attributes {
    description "SR topology scope attributes.";
    container sr {
        description
        "Containing SR attributes.";
        uses sr-cmn:srgb;
    } // sr
} // sr-topology-attributes
grouping information-source-attributes {
  description
  "The attributes identifying source that has provided the
  related information, and the source credibility.";
leaf information-source {
  type enumeration {
    enum "unknown" {
      description "The source is unknown.";
    }
    enum "locally-configured" {
      description "Configured entity.";
    }
    enum "ospfv2" {
      description "OSPFv2.";
    }
    enum "ospfv3" {
      description "OSPFv3.";
    }
    enum "isis" {
      description "ISIS.";
    }
    enum "bgp-1s" {
      description "BGP-LS.";
      reference
      "RFC 7752: North-Bound Distribution of Link-State and
      Traffic Engineering (TE) Information Using BGP";
    }
    enum "system-processed" {
      description "System processed entity.";
    }
    enum "other" {
      description "Other source.";
    }
  }
  config false;
  description
  "Indicates the type of the information source.";
}
leaf information-source-instance {
  type string;
  config false;
  description
  "The name indicating the instance of the information
  source.";
}
container information-source-state {
  config false;
  description
"The container contains state attributes related to
the information source."
leaf credibility-preference {
  type uint16;
  description
  "The preference value to calculate the traffic
  engineering database credibility value used for
tie-break selection between different
  information-source values.
  Higher value is more preferable.";
}
// information-source-attributes

// Operational state data
uses information-source-attributes;
// sr
}
// sr-node-attributes

// Functional state data
 uses sr-cmn:srgb;
 uses sr-cmn:srlb;
 uses sr-cmn:node-capabilities;
leaf msd {
  if-feature "msd";
  type uint8;
  description
  "Node MSD is the lowest MSD supported by the node.";
}
// sr
}
// sr-node-attributes

// Configuration state data
 uses sr-cmn:prefix-sid-attributes;
 uses sr-cmn:last-hop-behavior;
leaf is-local {
  type boolean;
  default false;
  description
  "true' if the SID is local.";
}
leaf is-node {

type boolean;
default false;
description
  "'true' if the Prefix-SID refers to the router identified
  by the prefix. Typically, the leaf 'is-node' (N-Flag)
  is set on Prefix-SIDs attached to a router loopback
  address.";
}
leaf is-readvertisement {
type boolean;
config false;
description
  "'true' if the prefix to which this Prefix-SID is attached,
  has been propagated by the router from another
topology by redistribution.";
}
} // sr
} // sr-node-prefix-attributes

grouping sr-link-attributes {
description "SR link scope attributes";
container sr {
presence "Presence indicates SR is enabled.";
description
  "Containing SR attributes.";
uses sr-cmn:sid-value-type;
leaf sid {
type uint32;
mandatory true;
description
  "Adjacency SID, which can be either IGP-Adjacency SID
  or BGP PeerAdj SID, depending on the context.";
}
leaf advertise-protection {
type enumeration {
enum "single" {
description
  "A single Adj-SID is associated
  with the adjacency and reflects
  the protection configuration.";
}
enum "dual" {
description
  "Two Adj-SIDs will be associated
  with the adjacency if interface
  is protected. In this case
  one will be enforced with
  backup flag set, the other
will be enforced to backup flag unset.
In case, protection is not configured,
a single Adj-SID will be advertised
with backup flag unset.

leaf is-local {
  type boolean;
  default false;
  description
  "'true' if the SID is local.";
}
leaf msd {
  if-feature "msd";
  type uint8;
  description
  "SID depth of the interface associated with the link.";
}
leaf address-family {
  type enumeration {
    enum "ipv4" {
      description
      "The Adj-SID refers to an adjacency with outgoing IPv4
      encapsulation.";
    }
    enum "ipv6" {
      description
      "The Adj-SID refers to an adjacency with outgoing IPv6
      encapsulation.";
    }
  }
  default "ipv4";
  description
  "This leaf defines the F-Flag (Address-Family flag) of the
  SID.";
}
leaf is-backup {
  type boolean;
  default false;
  description
  "'true' if the SID is a backup.";
}
leaf is-part-of-set {
type boolean;
default false;
description
  "true" if the SID is part of a set."
}
leaf is-persistent {
  type boolean;
  default true;
  description
  "true" if the SID is persistently allocated."
}
leaf is-on-lan {
  type boolean;
  default false;
  description
  "true" if on a lan."
}
  uses information-source-attributes;
    } // sr
  } // sr-tp-attributes
}</CODE ENDS>

5. IANA Considerations

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

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

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

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

Liu, et al.

Expires September 12, 2019

[Page 16]
6. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that 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., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

nw:network-types/l3t:l3-unicast-topology/sr-mpls
This subtree specifies the SR topology type. Modifying the configurations can make SR topology type invalid and cause interruption to all SR networks.

/nw/networks/nw:network/l3t:l3-topology-attributes/sr
This subtree specifies the topology-wide configurations, including the SRGB (Segment Routing Global Block). Modifying the configurations here can cause traffic disabled or rerouted in this topology and the connected topologies.
This subtree specifies the SR configurations for nodes. Modifying the configurations in this subtree can add, remove, or modify SR nodes, causing traffic disabled or rerouted in the specified nodes and the related TE topologies.

This subtree specifies the configurations for SR Adjacency Segments. Modifying the configurations in this subtree can add, remove, or modify SR Adjacency Segments causing traffic disabled or rerouted on the specified SR adjacencies, the related nodes, and the related SR topologies.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

Unauthorized access to this subtree can disclose the SR topology type.

Unauthorized access to this subtree can disclose the topology-wide configurations, including the SRGB (Segment Routing Global Block).

Unauthorized access to this subtree can disclose the operational state information of the SR nodes.

Unauthorized access to this subtree can disclose the operational state information of SR Adjacency Segments.

7. References

7.1. Normative References


7.2. Informative References


[RFC8346] Clemm, A., Medved, J., Varga, R., Liu, X.,
Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model
for Layer 3 Topologies", RFC 8346, DOI 10.17487/RFC8346,

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and
O. Dios, "YANG Data Model for Traffic Engineering (TE)
Topologies", draft-ietf-teas-yang-te-topo-19 (work in
progress), February 2019.

[I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Lindem, A., Sarkar, P., and J.
Tantsura, "YANG Data Model for Segment Routing", draft-
ietf-spring-sr-yang-12 (work in progress), February 2019.
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module ietf-sr-topology defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module, ietf-sr-topology-state, is defined as state model, which mirrors the module ietf-sr-topology defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

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

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

A.1. SR Topology State Module

<CODE BEGINS> file "ietf-sr-topology-state@2019-03-09.yang"
module ietf-sr-topology-state {
  yang-version 1.1;
  prefix "srt-s";

  import ietf-sr-topology {
    prefix "srt";
  }

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

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

  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }

  import ietf-segment-routing-common {
    prefix "sr-cmn";
    reference "I-D.ietf-spring-sr-yang: YANG Data Model for Segment Routing";

  }

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

contact
"WG Web: <http://tools.ietf.org/wg/teas/>
WG List: <mailto:teas@ietf.org>
Editor: Xufeng Liu
<mailto:xufeng.liu.ietf@gmail.com>
Editor: Igor Bryskin
<mailto:Igor.Bryskin@huawei.com>
Editor: Vishnu Pavan Beeram
<mailto:vbeeram@juniper.net>
Editor: Tarek Saad
<mailto:tsaad@cisco.com>
Editor: Himanshu Shah
<mailto:hshah@ciena.com>
Editor: Stephane Litkowski
<mailto:stephane.litkowski@orange.com>";

description
"YANG data model for representing operational state information
of Segment Routing Topologies, when NMDA is not supported.

Copyright (c) 2018 IETF Trust and the persons identified as
authors of the code. All rights reserved.

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

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

revision 2019-03-09 {
  description "Initial revision";
  reference
"RFC XXXX: YANG Data Model for SR and SR TE Topologies";
}

augment "/nw-s:networks/nw-s:network/nw-s:network-types/
+ "l3t-s:l3-unicast-topology" {
  description
  "Defines the SR topology type.";
  uses srt:sr-topology-type;
}

augment "/nw-s:networks/nw-s:network/
+ "l3t-s:l3-topology-attributes" {
  when "../nw-s:network-types/l3t-s:l3-unicast-topology/
+ "srt-s:sr-mpls" {
    description "Augment only for SR topology.";
  }
  description "Augment topology configuration";
  uses srt:sr-topology-attributes;
}

augment "/nw-s:networks/nw-s:network/nw-s:node/
+ "l3t-s:l3-node-attributes" {
  when "../../nw-s:network-types/l3t-s:l3-unicast-topology/
+ "srt-s:sr-mpls" {
    description "Augment only for SR topology.";
  }
  description "Augment node configuration.";
  uses srt:sr-node-attributes;
}

augment "/nw-s:networks/nw-s:network/nw-s:node/
+ "l3t-s:l3-node-attributes/l3t-s:prefix" {
  when "../../../nw-s:network-types/l3t-s:l3-unicast-topology/
+ "srt-s:sr-mpls" {
    description "Augment only for SR topology.";
  }
  description "Augment node prefix.";
  uses srt:sr-node-prefix-attributes;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/
+ "l3t-s:l3-link-attributes" {
  when "../../nw-s:network-types/l3t-s:l3-unicast-topology/
+ "srt-s:sr-mpls" {
    description "Augment only for SR topology.";
  }
  description "Augment link configuration";
  uses srt:sr-link-attributes;
grouping sr-topology-attributes {
    description "SR topology scope attributes."
    container sr {
        description "Containing SR attributes."
        uses sr-cmn:srgb;
    } // sr
} // sr-topology-attributes

Appendix B. Data Tree Example

This section contains an example of an instance data tree in the JSON encoding [RFC7951]. The example instantiates "ietf-sr-topology" for the topology that is depicted in the following diagram.

The corresponding instance data tree is depicted below. Note that some lines have been wrapped to adhere to the 72-character line limitation of RFCs.
Internet-Draft

YANG SR TE Topology

March 2019

{
"ietf-network:networks": {
"network": [
{
"network-types": {
"ietf-l3-unicast-topology:l3-unicast-topology": {
"ietf-sr-topology:sr-mpls": {}
}
},
"network-id": "sr-topo-example",
"ietf-l3-unicast-topology:l3-topology-attributes": {
"ietf-sr-topology:sr": {
"srgb": [
{
"lower-bound": 16000,
"upper-bound": 23999
}
]
}
},
"node": [
{
"node-id": "D1",
"ietf-network-topology:termination-point": [
{
"tp-id": "1-0-1",
"ietf-l3-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id": 101
}
},
{
"tp-id": "1-2-1",
"ietf-l3-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id": 121
}
},
{
"tp-id": "1-3-1",
"ietf-l3-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id": 131
}
}
],
"ietf-l3-unicast-topology:l3-node-attributes": {
"router-id": ["203.0.113.1"],
"prefix": [
{
"prefix": "203.0.113.1/32",

Liu, et al.

Expires September 12, 2019

[Page 25]


{"ietf-sr-topology:sr": { "start-sid": 101, "range": 1, "is-local": false, "is-node": true }},
{"ietf-sr-topology:sr": { "srgb": [ { "lower-bound": 16000, "upper-bound": 23999 } ], "srlb": [ { "lower-bound": 15000, "upper-bound": 15999 } ] }},
{"node-id": "D2", "ietf-network-topology:termination-point": [ { "tp-id": "2-0-1", "ietf-l3-unicast-topology:l3-termination-point-attributes": { "unnumbered-id": 201 } }, { "tp-id": "2-1-1", "ietf-l3-unicast-topology:l3-termination-point-attributes": { "unnumbered-id": 211 } }, { "tp-id": "2-3-1", "ietf-l3-unicast-topology:l3-termination-point-attributes": { "unnumbered-id": 231 } } ], "ietf-l3-unicast-topology:l3-node-attributes": { "router-id": ["203.0.113.2"]},
"prefix": [
  {
    "prefix": "203.0.113.2/32",
    "ietf-sr-topology:sr": {
      "start-sid": 102,
      "range": 1,
      "is-local": false,
      "is-node": true
    }
  }
],
"ietf-sr-topology:sr": {
  "srgb": [
    {
      "lower-bound": 16000,
      "upper-bound": 23999
    }
  ],
  "srlb": [
    {
      "lower-bound": 15000,
      "upper-bound": 15999
    }
  ]
}
],
"node-id": "D3",
"ietf-network-topology:termination-point": [
  {
    "tp-id": "3-1-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {
      "unnumbered-id": 311
    }
  },
  {
    "tp-id": "3-2-1",
    "ietf-l3-unicast-topology:l3-termination-point-attributes": {
      "unnumbered-id": 321
    }
  }
],
"ietf-l3-unicast-topology:l3-node-attributes": {
  "router-id": ["203.0.113.3"],
  "prefix": [
    "prefix": "203.0.113.3/32",
  ]}
"ietf-sr-topology:sr": {
   "start-sid": 101,
   "range": 1,
   "is-local": false,
   "is-node": true

},

"ietf-sr-topology:sr": {
   "srgb": [
      {
         "lower-bound": 16000,
         "upper-bound": 23999
      }
   ],
   "srlb": [
      {
         "lower-bound": 15000,
         "upper-bound": 15999
      }
   ]

},

"ietf-network-topology:link": [
   {
      "link-id": "D1,1-2-1,D2,2-1-1",
      "source": {
         "source-node": "D1",
         "source-tp": "1-2-1"
      },
      "destination": {
         "dest-node": "D2",
         "dest-tp": "2-1-1"
      },
      "ietf-l3-unicast-topology:l3-link-attributes": {
         "metric1": "100",
         "ietf-sr-topology:sr": {
            "sid": 121,
            "is-local": true
         }
      }
   },
   {
      "link-id": "D2,2-1-1,D1,1-2-1",
      "source": {
         "source-node": "D2",
         "source-tp": "2-1-1"
      },
      "destination": {
         "dest-node": "D1",
         "dest-tp": "1-2-1"
      },
      "ietf-l3-unicast-topology:l3-link-attributes": {
         "metric1": "100",
         "ietf-sr-topology:sr": {
            "sid": 121,
            "is-local": true
         }
      }
   }
]
"source-tp": "2-1-1"
},
"destination": {
"dest-node": "D1",
"dest-tp": "1-2-1"
},
"ietf-l3-unicast-topology:l3-link-attributes": {
"metric1": "100",
"ietf-sr-topology:sr": {
"sid": 211,
"is-local": true
}
}
},
{
"link-id": "D1,1-3-1,D3,3-1-1",
"source": {
"source-node": "D1",
"source-tp": "1-3-1"
},
"destination": {
"dest-node": "D3",
"dest-tp": "3-1-1"
},
"ietf-l3-unicast-topology:l3-link-attributes": {
"metric1": "100",
"ietf-sr-topology:sr": {
"sid": 131,
"is-local": true
}
}
},
{
"link-id": "D3,3-1-1,D1,1-3-1",
"source": {
"source-node": "D3",
"source-tp": "3-1-1"
},
"destination": {
"dest-node": "D1",
"dest-tp": "1-3-1"
},
"ietf-l3-unicast-topology:l3-link-attributes": {
"metric1": "100",
"ietf-sr-topology:sr": {
"sid": 311,
"is-local": true
}
}
```json
{
  "link-id": "D2,2-3-1,D3,3-2-1",
  "source": {
    "source-node": "D2",
    "source-tp": "2-3-1"
  },
  "destination": {
    "dest-node": "D3",
    "dest-tp": "3-2-1"
  },
  "ietf-l3-unicast-topology:l3-link-attributes": {
    "metric1": "100",
    "ietf-sr-topology:sr": {
      "sid": 231,
      "is-local": true
    }
  }
},
{
  "link-id": "D3,3-2-1,D2,2-3-1",
  "source": {
    "source-node": "D3",
    "source-tp": "3-2-1"
  },
  "destination": {
    "dest-node": "D2",
    "dest-tp": "2-3-1"
  },
  "ietf-l3-unicast-topology:l3-link-attributes": {
    "metric1": "100",
    "ietf-sr-topology:sr": {
      "sid": 321,
      "is-local": true
    }
  }
}
```
Appendix C. Contributors

Jeff Tantsura
Email: jefftant.ietf@gmail.com

Yingzhen Qu
Email: yingzhen.qu@huawei.com

Authors' Addresses

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

Igor Bryskin
Huawei Technologies
EMail: Igor.Bryskin@huawei.com

Vishnu Pavan Beeram
Juniper Networks
EMail: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc
EMail: tsaad@cisco.com

Himanshu Shah
Ciena
EMail: hshah@ciena.com

Stephane Litkowski
Orange Business Service
EMail: stephane.litkowski@orange.com

Abstract

This document defines a YANG data model for the configuration and management of Traffic Engineering (TE) interfaces, tunnels and Label Switched Paths (LSPs). The model is divided into YANG modules that classify data into generic, device-specific, technology agnostic, and technology-specific elements.

This model covers data for configuration, operational state, remote procedural calls, and event notifications.

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 October 11, 2019.
1. Introduction

YANG [RFC6020] and [RFC7950] is a data modeling language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG has proved relevant beyond its initial confines, as bindings to other interfaces (e.g. RESTCONF [RFC8040]) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document describes YANG data model for TE Tunnels, Label Switched Paths (LSPs) and TE interfaces and covers data applicable to generic or device-independent, device-specific, and Multiprotocol Label Switching (MPLS) technology specific.

The document describes a high-level relationship between the modules defined in this document, as well as other external protocol YANG modules. The TE generic YANG data model does not include any data specific to a signaling protocol. It is expected other data plane technology model(s) will augment the TE generic YANG data model.

Also, it is expected other YANG module(s) that model TE signaling protocols, such as RSVP-TE ([RFC3209], [RFC3473]), or Segment-Routing TE (SR-TE) will augment the TE generic YANG module.

1.1. Terminology

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

The terminology for describing YANG data models is found in [RFC7950].

1.2. Prefixes in Data Node Names

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>rt-types</td>
<td>ietf-routing-types</td>
<td>[RFC8294]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>this document</td>
</tr>
<tr>
<td>te-dev</td>
<td>ietf-te-device</td>
<td>this document</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[I-D.ietf-teas-yang-te-types]</td>
</tr>
<tr>
<td>te-mpls-types</td>
<td>ietf-te-mpls-types</td>
<td>[I-D.ietf-teas-yang-te-types]</td>
</tr>
</tbody>
</table>

1.3. TE Technology Models

This document describes the TE generic YANG data model that is independent of any dataplane technology. One of the design objectives is to allow specific data plane technology models to reuse the TE generic data model and possibly augment it with technology specific data.

The elements of the TE generic YANG data model, including TE tunnels, LSPs, and interfaces have leaf(s) that identify the technology layer where they reside. For example, the LSP encoding type can identify the technology associated with a TE tunnel or LSP.

Also, the TE generic YANG data model does not cover signaling protocol data. This is expected to be covered by augmentations defined in other document(s).

1.4. State Data Organization

The Network Management Datastore Architecture (NMDA) [RFC8342] addresses modeling state data for ephemeral objects. This draft adopts the NMDA proposal for configuration and state data representation as per IETF guidelines for new IETF YANG models.

2. Model Overview

The data model(s) defined in this document cover core TE features that are commonly supported across different vendor implementations. The support of extended or vendor specific TE feature(s) is expected to be in augmentations to the base model defined in this document.
2.1. Module(s) Relationship

The TE generic YANG data model defined in "ietf-te.yang" covers the building blocks that are device independent and agnostic of any specific technology or control plane instances. The TE device model defined in "ietf-te-device.yang" augments the TE generic YANG data model and covers data that is specific to a device – for example, attributes of TE interfaces, or TE timers that are local to a TE node.

The TE data model for specific instances of data plane technology exist in a separate YANG module(s) that augment the TE generic YANG data model. For example, the MPLS-TE module "ietf-te-mpls.yang" is defined in another document and augments the TE generic model as shown in Figure 1.

The TE data model for specific instances of signaling protocol are outside the scope of this document and are defined in other documents. For example, the RSVP-TE YANG model augmentation of the TE model is covered in [I-D.ietf-teas-yang-rsvp].
Figure 1: Relationship of TE module(s) with other signaling protocol modules

Figure 2: Relationship between generic and technology specific TE types modules

*: not in this document, shown for illustration only
2.2. Design Considerations

The following design considerations are taken into account with respect data organization:

- reusable TE data types that are data plane independent are grouped in the TE generic types module "ietf-te-types.yang" defined in [I-D.ietf-teas-yang-te-types]

- reusable TE data types that are data plane specific are defined in a data plane type module, e.g. "ietf-te-packet-types.yang" as defined in [I-D.ietf-teas-yang-te-types]. Other data plane types are expected to be defined in separate module(s) as shown in Figure 2.

- The TE generic YANG data model "ietf-te" contains device independent data and can be used to model data off a device (e.g. on a controller). The device-specific TE data is defined in module "ietf-te-device" as shown in Figure 1.

- In general, minimal elements in the model are designated as "mandatory" to allow freedom to vendors to adapt the data model to their specific product implementation.

- This model declares a number of TE functions as features that can be optionally supported.

2.3. Model Tree Diagram

Figure 3 shows the tree diagram of the TE YANG model defined in modules: ietf-te.yang, and ietf-te-device.yang.

```
module: ietf-te
  +--rw te!
    +--rw globals
      +--rw named-admin-groups
        +--rw named-admin-group* [name]
          +--rw name          string
          +--rw bit-position? uint32
      +--rw named-srlgs
        +--rw named-srlg* [name] {te-types:named-srlg-groups}?
          +--rw name          string
          +--rw group?        te-types:srlg
          +--rw cost?         uint32
      +--rw named-path-constraints
        +--rw named-path-constraint* [name]
          {te-types:named-path-constraints}?
          +--rw name          string
```
```yaml
++-rw te-bandwidth
  ++-rw (technology)?
  |    ++-:(generic)
  |    |    ++-rw generic? te-bandwidth
  ++-rw link-protection? identityref
  ++-rw setup-priority? uint8
  ++-rw hold-priority? uint8
  ++-rw signaling-type? identityref
  ++-rw path-metric-bounds
  |    ++-rw path-metric-bound* [metric-type]
  |    |    ++-rw metric-type identityref
  |    |    ++-rw upper-bound? uint64
  ++-rw path-affinities-values
  |    ++-rw path-affinities-value* [usage]
  |    |    ++-rw usage identityref
  |    |    ++-rw value? admin-groups
  ++-rw path-affinity-names
  |    ++-rw path-affinity-name* [usage]
  |    |    ++-rw usage identityref
  |    |    ++-rw affinity-name* [name]
  |    |    |    ++-rw name string
  ++-rw path-srlgs-lists
  |    ++-rw path-srlgs-list* [usage]
  |    |    ++-rw usage identityref
  |    |    ++-rw values* srlg
  ++-rw path-srlgs-names
  |    ++-rw path-srlgs-name* [usage]
  |    |    ++-rw usage identityref
  |    |    ++-rw names* string
  ++-rw disjointness?
  |    ++-:(te-path-disjointness)
  ++-rw explicit-route-objects-always
  |    ++-rw route-object-exclude-always* [index]
  |    |    ++-rw index uint32
  |    |    ++-:(numbered-node-hop)
  |    |    |    ++-rw numbered-node-hop
  |    |    |    |    ++-rw node-id te-node-id
  |    |    |    |    ++-rw hop-type? te-hop-type
  |    |    |    ++-:(numbered-link-hop)
  |    |    |    |    ++-rw numbered-link-hop
  |    |    |    |    |    ++-rw link-tp-id te-tp-id
  |    |    |    |    |    ++-rw hop-type? te-hop-type
  |    |    |    |    |    ++-rw direction? te-link-direction
  |    |    |    ++-:(unnumbered-link-hop)
  |    |    |    |    ++-rw unnumbered-link-hop
  |    |    |    |    |    ++-rw link-tp-id te-tp-id
  |    |    |    |    |    ++-rw node-id te-node-id
```
++-rw srlg
    ++-rw srlg?  uint32
++-rw shared-resources-tunnels
    ++-rw lsp-shared-resources-tunnel*  tunnel-ref
++-rw path-in-segment!
    ++-rw label-restrictions
        ++-rw label-restriction* [index]
            ++-rw restriction?  enumeration
            ++-rw index  uint32
        ++-rw label-start
            ++-rw te-label
                ++-rw (technology)?
                    ++-:(generic)
                    ++-rw generic?
                        rt-types:generalized-label
                ++-rw direction?  te-label-direction
        ++-rw label-end
            ++-rw te-label
                ++-rw (technology)?
                    ++-:(generic)
                    ++-rw generic?
                        rt-types:generalized-label
                ++-rw direction?  te-label-direction
        ++-rw label-step
            ++-rw (technology)?
                ++-:(generic)
                    ++-rw generic?  int32
            ++-rw range-bitmap?  yang:hex-string
++-rw path-out-segment!
    ++-rw label-restrictions
        ++-rw label-restriction* [index]
            ++-rw restriction?  enumeration
            ++-rw index  uint32
        ++-rw label-start
            ++-rw te-label
                ++-rw (technology)?
                    ++-:(generic)
                    ++-rw generic?
                        rt-types:generalized-label
                ++-rw direction?  te-label-direction
        ++-rw label-end
            ++-rw te-label
                ++-rw (technology)?
                    ++-:(generic)
                    ++-rw generic?
                        rt-types:generalized-label
                ++-rw direction?  te-label-direction
        ++-rw label-step
++--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 te-topology-identifier
    ++--rw provider-id? te-global-id
    ++--rw client-id? te-global-id
    ++--rw topology-id? te-topology-id
++--rw te-bandwidth
    ++--rw (technology)?
        ++--:(generic)
            ++--rw generic? te-bandwidth
        ++--rw link-protection? identityref
    ++--rw setup-priority? uint8
    ++--rw hold-priority? uint8
    ++--rw signaling-type? identityref
++--rw dependency-tunnels
    ++--rw dependency-tunnel* [name]
        ++--rw name
            | -> ../../../../../tunnels/tunnel/name
        ++--rw encoding? identityref
        ++--rw switching-type? identityref
++--rw hierarchical-link
    ++--rw local-te-node-id? te-types:te-node-id
    ++--rw local-te-link-tp-id? te-types:te-tp-id
    ++--rw remote-te-node-id? te-types:te-node-id
    ++--rw te-topology-identifier
        ++--rw provider-id? te-global-id
        ++--rw client-id? te-global-id
        ++--rw topology-id? te-topology-id
++--rw p2p-primary-paths
    ++--rw p2p-primary-path* [name]
        ++--rw name
        ++--rw path-setup-protocol? identityref
        ++--rw path-computation-method? identityref
        ++--rw path-computation-server?
            | inet:ip-address
        ++--rw compute-only? empty
        ++--rw use-path-computation? boolean
        ++--rw lockdown? empty
        ++--ro path-scope? identityref
        ++--rw optimizations
            ++--rw (algorithm)?
                ++--:(metric) (path-optimization-metric)?
                    ++--rw optimization-metric* [metric-type]
                        ++--rw metric-type identityref
---rw weight?
  uint8
---rw explicit-route-exclude-objects
  +--rw route-object-exclude-object*
    [index]
    +--rw index
      |  uint32
    +--rw (type)?
      +--:(numbered-node-hop)
        +--rw numbered-node-hop
          +--rw node-id
            |  te-node-id
          +--rw hop-type?
            |  te-hop-type
        +--:(numbered-link-hop)
          +--rw numbered-link-hop
            +--rw link-tp-id
              |  te-tp-id
            +--rw hop-type?
              |  te-hop-type
            +--rw direction?
              |  te-link-direction
        +--:(unnumbered-link-hop)
          +--rw unnumbered-link-hop
            +--rw link-tp-id
              |  te-tp-id
            +--rw node-id
              |  te-node-id
            +--rw hop-type?
              |  te-hop-type
            +--rw direction?
              |  te-link-direction
        +--:(as-number)
          +--rw as-number-hop
            +--rw as-number
              |  inet:as-number
            +--rw hop-type?
              |  te-hop-type
        +--:(label)
          +--rw label-hop
            +--rw te-label
              +--rw (technology)?
                +--:(generic)
                  +--rw generic?
                    rt-types:generalized-label
                    +--rw direction?
                      |  te-label-direction
            +--:(srlg)
>--rw tiebreaker* [tiebreaker-type]
   +--rw tiebreaker-type identityref
++--:(objective-function)
   [path-optimization-objective-function]?
   +--rw objective-function
   +--rw objective-function-type?
      identityref
++--rw preference? uint8
++--rw k-requested-paths? uint8
++--rw named-path-constraint? leafref
   [te-types:named-path-constraints]?
++--rw te-bandwidth
   ++--rw (technology)?
      ++--:(generic)
         +--rw generic? te-bandwidth
++--rw link-protection? identityref
++--rw setup-priority? uint8
++--rw hold-priority? uint8
++--rw signaling-type? identityref
++--rw path-metric-bounds
   ++--rw path-metric-bound* [metric-type]
      ++--rw metric-type identityref
      ++--rw upper-bound? uint64
++--rw path-affinities-values
   ++--rw path-affinities-value* [usage]
      ++--rw usage identityref
      ++--rw value? admin-groups
++--rw path-affinity-names
   ++--rw path-affinity-name* [usage]
      ++--rw usage identityref
      ++--rw affinity-name* [name]
         ++--rw name string
++--rw path-srlgs-lists
   ++--rw path-srlgs-list* [usage]
      ++--rw usage identityref
      ++--rw values* srlg
++--rw path-srlgs-names
   ++--rw path-srlgs-name* [usage]
      ++--rw usage identityref
      ++--rw names* string
++--rw disjointness?
   te-path-disjointness
++--rw explicit-route-objects-always
   ++--rw route-object-exclude-always* [index]
      ++--rw index uint32
      ++--rw (type)?
         ++--:(numbered-node-hop)
            ++--rw numbered-node-hop
+-rw (technology)?
  |  +--:(generic)
  |     +-rw generic?
  |          rt-types:generalized-label
  |     ++-rw direction?
  |          te-label-direction
+-rw label-end
  +-rw te-label
     +-rw (technology)?
       |  +--:(generic)
       |     +-rw generic?
       |          rt-types:generalized-label
       |     ++-rw direction?
       |          te-label-direction
+-rw label-step
     +-rw (technology)?
       |  +--:(generic)
       |     +-rw generic?  int32
     ++-rw range-bitmap?  yang:hex-string
+-ro computed-paths-properties
     +-ro computed-path-properties* [k-index]
       +-ro k-index  uint8
     +-ro path-properties
         +-ro path-metric* [metric-type]
             |  +-ro metric-type  identityref
             |  +-ro accumulative-value?  uint64
         +-ro path- affinities-values
             +-ro path- affinities-value* [usage]
                 |  +-ro usage  identityref
                 |  +-ro value?  admin-groups
         +-ro path-affinity-names
             +-ro path-affinity-name* [usage]
                 |  +-ro usage  identityref
                 |  +-ro affinity-name* [name]
                     |     +-ro name  string
         +-ro path-srlgs-lists
             +-ro path-srlgs-list* [usage]
                 |  +-ro usage  identityref
                 |  +-ro values*  srlg
         +-ro path-srlgs-names
             +-ro path-srlgs-name* [usage]
                 |  +-ro usage  identityref
                 |  +-ro names*  string
         +-ro path-route-objects
             +-ro path-computed-route-object* [index]
                 |     +-ro index  uint32
                 |     +-ro (type)?
++-ro error-link-id?              te-types:te-tp-id
++-ro lsp-id?                    uint16
++-ro lsp* [lsp-id]
    ++-ro lsp-provisioning-error-infos
        ++-ro lsp-provisioning-error-info* []
            ++-ro error-description?     string
            |  yang:date-and-time
            ++-ro error-node-id?
            |  te-types:te-node-id
            ++-ro error-link-id?
            |  te-types:te-tp-id
    ++-ro source?
        |  te-types:te-node-id
    ++-ro destination?
        |  te-types:te-node-id
    ++-ro tunnel-id?
        |  uint16
    ++-ro lsp-id
        |  uint16
    ++-ro extended-tunnel-id?
        |  yang:dotted-quad
    ++-ro operational-state?
        |  identityref
    ++-ro path-setup-protocol?
        |  identityref
    ++-ro origin-type?
        |  enumeration
    ++-ro lsp-resource-status?
        |  enumeration
    ++-ro lockout-of-normal?
        |  boolean
    ++-ro freeze?
        |  boolean
    ++-ro lsp-protection-role?
        |  enumeration
    ++-ro lsp-protection-state?
        |  identityref
    ++-ro protection-group-ingress-node-id?
        |  te-types:te-node-id
    ++-ro protection-group-egress-node-id?
        |  te-types:te-node-id
    ++-ro lsp-shared-resources-tunnel?
        |  tunnel-ref
    ++-ro lsp-record-route-information
        ++-ro lsp-record-route-information* [index]
            ++-ro index          uint32
---ro (type)?
  ---: (numbered-node-hop)
    ---ro numbered-node-hop
      ---ro node-id  te-node-id
      ---ro flags*
        path-attribute-flags
  ---: (numbered-link-hop)
    ---ro numbered-link-hop
      ---ro link-tp-id  te-tp-id
      ---ro flags*
        path-attribute-flags
  ---: (unnumbered-link-hop)
    ---ro unnumbered-link-hop
      ---ro link-tp-id  te-tp-id
      ---ro node-id?  te-node-id
      ---ro flags*
        path-attribute-flags
  ---: (label)
    ---ro label-hop
      ---ro te-label
        ---ro (technology)?
          ---: (generic)
            ---ro generic?
              rt-types:generalized-label
                ---ro direction?
                  te-label-direction
            ---ro flags*
              path-attribute-flags
      ---ro path-properties
        ---ro path-metric* [metric-type]
          ---ro metric-type  identityref
          ---ro accumulative-value?  uint64
        ---ro path-affinities-values
          ---ro path-affinities-value* [usage]
            ---ro usage  identityref
            ---ro value?  admin-groups
          ---ro path-affinity-names
            ---ro path-affinity-name* [usage]
              ---ro usage  identityref
              ---ro affinity-name* [name]
                ---ro name  string
          ---ro path-srlgs-lists
            ---ro path-srlgs-list* [usage]
              ---ro usage  identityref
              ---ro values*  srlg
            ---ro path-srlgs-names
              ---ro path-srlgs-name* [usage]
                ---ro usage  identityref
---ro names* string

---ro path-route-objects

  ---ro path-computed-route-object* [index]
   |  ---ro index
   |   |       uint32
   |  ---ro (type)?
   |    |       ---ro (numbered-node-hop)
   |    |       ---ro numbered-node-hop
   |    |         |       ---ro node-id  te-node-id
   |    |         |       ---ro hop-type?  te-hop-type
   |    |       ---ro (numbered-link-hop)
   |    |       ---ro numbered-link-hop
   |    |         |       ---ro link-tp-id  te-tp-id
   |    |         |       ---ro hop-type?  te-hop-type
   |    |         |       ---ro direction?  te-link-direction
   |    |       ---ro (unnumbered-link-hop)
   |    |       ---ro unnumbered-link-hop
   |    |         |       ---ro link-tp-id  te-tp-id
   |    |         |       ---ro node-id  te-node-id
   |    |         |       ---ro hop-type?  te-hop-type
   |    |         |       ---ro direction?  te-link-direction
   |    |       ---ro (as-number)
   |    |       ---ro as-number-hop
   |    |         |       ---ro as-number
   |    |         |       |       inet:as-number
   |    |         |       ---ro hop-type?  te-hop-type
   |    |       ---ro (label)
   |    |       ---ro label-hop
   |    |         |       ---ro te-label
   |    |         |       |       (technology)?
   |    |         |       |       |       ---ro generic?
   |    |         |       |       |         rt-types:generalized-label
   |    |         |       |       ---ro direction?  te-label-direction
   |    |         |       ---ro shared-resources-tunnels
   |    |         |       ---ro lsp-shared-resources-tunnel*
   |    |         |         tunnel-ref
   |    |         |       ---ro te-dev: lsp-timers
   |    |         |       +=ro te-dev:life-time?  uint32
   |    |         |       +=ro te-dev:time-to-install?  uint32
   |    |         |       +=ro te-dev:time-to-destroy?  uint32
+---ro te-dev:downstream-info
  +-ro te-dev:nhop?
   |  inet:ip-address
  +-ro te-dev:outgoing-interface?
   |  if:interface-ref
  +-ro te-dev:neighbor?
   |  inet:ip-address
  +-ro te-dev:label?
   |  rt-types:generalized-label
  +-ro te-dev:upstream-info
   +-ro te-dev:nhop?  inet:ip-address
   +-ro te-dev:neighbor?  inet:ip-address
   +-ro te-dev:label?
   |  rt-types:generalized-label
  ---rw p2p-primary-reverse-path
   ---rw name?  string
   ---rw path-setup-protocol?  identityref
   ---rw path-computation-method?  identityref
   ---rw path-computation-server?
    |  inet:ip-address
   ---rw compute-only?  empty
   ---rw use-path-computation?  boolean
   ---rw lockdown?  empty
   ---ro path-scope?  identityref
  ---rw optimizations
   ---rw (algorithm)?
    |  ---:(metric) {path-optimization-metric}?  [metric-type]
     |  ---rw metric-type  identityref
     |  ---rw weight?  uint8
     |  ---rw explicit-route-exclude-objects
      |  ---rw route-object-exclude-object*  [index]
       |  ---rw index  uint32
     |  ---rw (type)?
      |  ---:(numbered-node-hop)
       |  ---rw numbered-node-hop
        |  ---rw node-id  te-node-id
        |  ---rw hop-type?  te-hop-type
      |  ---:(numbered-link-hop)
       |  ---rw numbered-link-hop
        |  ---rw link-tp-id  te-tp-id
---rw hop-type?
|     te-hop-type
---rw direction?
|     te-link-direction
++-:(unnumbered-link-hop)
  ---rw unnumbered-link-hop
    ---rw link-tp-id
      |     te-tp-id
    ---rw node-id
      |     te-node-id
    ---rw hop-type?
      |     te-hop-type
    ---rw direction?
      te-link-direction
++-:(as-number)
  ---rw as-number-hop
    ---rw as-number
      inet:as-number
    ---rw hop-type?
      te-hop-type
++-:(label)
  ---rw label-hop
    ---rw te-label
      ---rw (technology)?
        |     +--:(generic)
        ---rw generic?
        rt-types:generalized-label
    ---rw direction?
      te-label-direction
  ---rw tiebreakers
    ---rw tiebreaker* [tiebreaker-type]
      ---rw tiebreaker-type
        identityref
++-:(objective-function)
  {path-optimization-objective-function}?
    ---rw objective-function
      ---rw objective-function-type?
        identityref
    ---rw named-path-constraint? leafref
      {te-types:named-path-constraints}?
  ---rw te-bandwidth
    ---rw (technology)?
      |     +--:(generic)
      ---rw generic? te-bandwidth
    ---rw link-protection? identityref
    ---rw setup-priority? uint8
    ---rw hold-priority? uint8
    ---rw signaling-type? identityref
---(label)
  +--rw label-hop
    +--rw te-label
      +--rw (technology)?
        +--:(generic)
          +--rw generic?
            rt-types:generalized-label
      +--rw direction?
        te-label-direction
  +--rw route-object-include-exclude* [index]
    +--rw explicit-route-usage?
      identityref
    +--rw index uint32
  +--rw (type)?
    +--:(numbered-node-hop)
      +--rw numbered-node-hop
        +--rw node-id te-node-id
        +--rw hop-type? te-hop-type
    +--:(numbered-link-hop)
      +--rw numbered-link-hop
        +--rw link-tp-id te-tp-id
        +--rw hop-type? te-hop-type
        +--rw direction?
          te-link-direction
    +--:(unnumbered-link-hop)
      +--rw unnumbered-link-hop
        +--rw link-tp-id te-tp-id
        +--rw node-id te-node-id
        +--rw hop-type? te-hop-type
        +--rw direction?
          te-link-direction
    +--:(as-number)
      +--rw as-number-hop
        +--rw as-number inet:as-number
        +--rw hop-type? te-hop-type
    +--:(label)
      +--rw label-hop
        +--rw te-label
          +--rw (technology)?
            +--:(generic)
              +--rw generic?
                rt-types:generalized-label
          +--rw direction?
            te-label-direction
    +--:(srlg)
      +--rw srlg
        +--rw srlg? uint32
      +--rw shared-resources-tunnels
---rw lsp-shared-resources-tunnel*  tunnel-ref
  ---rw path-in-segment!
  +--rw label-restrictions
     +--rw label-restriction* [index]
        +--rw restriction?  enumeration
        +--rw index        uint32
        +--rw label-start
           +--rw te-label
              +--rw (technology)?
                 +--:(generic)
                    +--rw generic?  rt-types:generalized-label
              +--rw direction?  te-label-direction
        +--rw label-end
           +--rw te-label
              +--rw (technology)?
                 +--:(generic)
                    +--rw generic?  rt-types:generalized-label
              +--rw direction?  te-label-direction
        +--rw label-step
           +--rw (technology)?
                 +--:(generic)
                    +--rw generic?  int32
        +--rw range-bitmap?  yang:hex-string
  +--rw path-out-segment!
     +--rw label-restrictions
        +--rw label-restriction* [index]
           +--rw restriction?  enumeration
           +--rw index        uint32
           +--rw label-start
              +--rw te-label
                 +--rw (technology)?
                    +--:(generic)
                       +--rw generic?  rt-types:generalized-label
              +--rw direction?  te-label-direction
           +--rw label-end
              +--rw te-label
                 +--rw (technology)?
                    +--:(generic)
                       +--rw generic?  rt-types:generalized-label
              +--rw direction?  te-label-direction
+++rw label-step
  +++rw (technology)?
    +++:(generic)
      +++rw generic? int32
      +++rw range-bitmap? yang:hex-string
  +++ro computed-paths-properties
    +++ro computed-path-properties* [k-index]
      +++ro k-index uint8
      +++ro path-properties
        +++ro path-metric* [metric-type]
          +++ro metric-type
            | identityref
          +++ro accumulative-value? uint64
        +++ro path-affinities-values
          +++ro path-affinities-value* [usage]
            +++ro usage identityref
            +++ro value? admin-groups
        +++ro path-affinity-names
          +++ro path-affinity-name* [usage]
            +++ro usage identityref
            +++ro affinity-name* [name]
              +++ro name string
        +++ro path-srlgs-lists
          +++ro path-srlgs-list* [usage]
            +++ro usage identityref
            +++ro values* srlg
        +++ro path-srlgs-names
          +++ro path-srlgs-name* [usage]
            +++ro usage identityref
            +++ro names* string
      +++ro path-route-objects
        +++ro path-computed-route-object* [index]
          +++ro index uint32
        +++ro (type)?
          +++:(numbered-node-hop)
            +++ro numbered-node-hop
              +++ro node-id
                | te-node-id
              +++ro hop-type?
                | te-hop-type
          +++:(numbered-link-hop)
            +++ro numbered-link-hop
              +++ro link-tp-id
                | te-tp-id
              +++ro hop-type?
                | te-hop-type
---ro direction?
    te-link-direction
---:(unnumbered-link-hop)
  ---ro unnumbered-link-hop
    ---ro link-tp-id
      te-tp-id
    ---ro node-id
      te-node-id
    ---ro hop-type?
      te-hop-type
  ---ro direction?
    te-link-direction
---:(as-number)
  ---ro as-number-hop
    ---ro as-number
      inet:as-number
    ---ro hop-type?
      te-hop-type
---:(label)
  ---ro label-hop
    ---ro te-label
      ---ro (technology)?
      ---:(generic)
        ---ro generic?
          rt-types:generalized-label
          ---ro direction?
            te-label-direction
    ---ro shared-resources-tunnels
      ---ro lsp-shared-resources-tunnel*
        tunnel-ref
      ---ro computed-path-error-infos
        ---ro computed-path-error-info* []
          ---ro error-description? string
          ---ro error-timestamp?
            yang:date-and-time
          ---ro error-reason?
            identityref
        ---ro lsp-provisioning-error-infos
          ---ro lsp-provisioning-error-info* []
            ---ro error-description? string
            ---ro error-timestamp?
              yang:date-and-time
            ---ro error-node-id?
              te-types:te-node-id
            ---ro error-link-id?
              te-types:te-tp-id
            ---ro lsp-id?
              uint16
      ---ro lsps
        ---ro lsp* [lsp-id]
          ---ro lsp-provisioning-error-infos
++ro lsp-provisioning-error-info* []
  +--ro error-description?    string
  +--ro error-timestamp?
     |     yang:date-and-time
  +--ro error-node-id?
     |     te-types:te-node-id
  +--ro error-link-id?
     |     te-types:te-tp-id
++ro source?
  |     te-types:te-node-id
++ro destination?
  |     te-types:te-node-id
++ro tunnel-id?
  |     uint16
++ro lsp-id
  |     uint16
++ro extended-tunnel-id?
  |     yang:dotted-quad
++ro operational-state?
  |     identityref
++ro path-setup-protocol?
  |     identityref
++ro origin-type?
  |     enumeration
++ro lsp-resource-status?
  |     enumeration
++ro lockout-of-normal?
  |     boolean
++ro freeze?
  |     boolean
++ro lsp-protection-role?
  |     enumeration
++ro lsp-protection-state?
  |     identityref
++ro protection-group-ingress-node-id?
  |     te-types:te-node-id
++ro protection-group-egress-node-id?
  |     te-types:te-node-id
++ro lsp-shared-resources-tunnel?
  |     tunnel-ref
++ro lsp-record-route-information
  +--ro lsp-record-route-information* [index]
     |     +--ro index
     |     |     uint32
     |     +--ro (type)?
     |     |     +--:(numbered-node-hop)
     |     |     |     +--ro numbered-node-hop
---ro node-id    te-node-id
---ro flags*
    path-attribute-flags
  :numbered-link-hop
---ro numbered-link-hop
    ---ro link-tp-id    te-tp-id
    ---ro flags*
    path-attribute-flags
  :unnumbered-link-hop
    ---ro link-tp-id    te-tp-id
    ---ro node-id?
        te-node-id
    ---ro flags*
    path-attribute-flags
  :label
---ro label-hop
    ---ro te-label
        ---ro (technology)?
        :generic
            ---ro generic?
            rt-types:generalized-label
        ---ro direction?
            te-label-direction
        ---ro flags*
            path-attribute-flags
    ---ro path-properties
        ---ro path-metric* [metric-type]
            ---ro metric-type
                identityref
            ---ro accumulative-value?   uint64
        ---ro path-affinities-values
            ---ro path-affinities-value* [usage]
                ---ro usage     identityref
                ---ro value?   admin-groups
        ---ro path-affinity-names
            ---ro path-affinity-name* [usage]
                ---ro usage     identityref
                ---ro affinity-name* [name]
                    ---ro name    string
        ---ro path-srlgs-lists
            ---ro path-srlgs-list* [usage]
                ---ro usage     identityref
                ---ro values*   srlg
        ---ro path-srlgs-names
            ---ro path-srlgs-name* [usage]
                ---ro usage     identityref
                ---ro names*   string

```yang
++-ro path-route-objects
  ++-ro path-computed-route-object*
      [index]
      ++-ro index
          uint32
      ++-ro (type)?
          +--:(numbered-node-hop)
              ++-ro numbered-node-hop
                  ++-ro node-id
                      te-node-id
                  ++-ro hop-type?
                      te-hop-type
          +--:(numbered-link-hop)
              ++-ro numbered-link-hop
                  ++-ro link-tp-id
                      te-tp-id
                  ++-ro hop-type?
                      te-hop-type
                  ++-ro direction?
                      te-link-direction
          +--:(unnumbered-link-hop)
              ++-ro unnumbered-link-hop
                  ++-ro link-tp-id
                      te-tp-id
                  ++-ro node-id
                      te-node-id
                  ++-ro hop-type?
                      te-hop-type
                  ++-ro direction?
                      te-link-direction
          +--:(as-number)
              ++-ro as-number-hop
                  ++-ro as-number
                      inet:as-number
                  ++-ro hop-type?
                      te-hop-type
          +--:(label)
              ++-ro label-hop
                  ++-ro te-label
                      +-- ro (technology)?
                          +--:(generic)
                              ++-ro generic?
                              rt-types:generalized-label
                      ++-ro direction?
                          te-label-direction
  ++-ro shared-resources-tunnels
      ++-ro lsp-shared-resources-tunnel*
          tunnel-ref
```

+-rw p2p-secondary-reverse-path
  +-rw secondary-path? leafref
  +-rw path-setup-protocol? identityref
+-rw candidate-p2p-secondary-paths
  +-rw candidate-p2p-secondary-path* [secondary-path]
    +-rw secondary-path leafref
    +-rw path-setup-protocol? identityref
    +-ro active? boolean
+-rw p2p-secondary-paths
  +-rw p2p-secondary-path* [name]
    +-rw name string
    +-rw path-setup-protocol? identityref
    +-rw path-computation-method? identityref
    +-rw path-computation-server?
      inet:ip-address
    +-rw compute-only? empty
    +-rw use-path-computation? boolean
    +-rw lockdown? empty
    +-ro path-scope? identityref
  +-rw optimizations
    +-rw (algorithm)?
      +-:(metric) [path-optimization-metric]?
        +-rw optimization-metric* [metric-type]
          +-rw metric-type identityref
          +-rw weight? uint8
        +-rw explicit-route-exclude-objects
          +-rw route-object-exclude-object* [index]
            +-rw index uint32
        +-rw (type)?
          +-:(numbered-node-hop)
            +-rw numbered-node-hop
              +-rw node-id te-node-id
              +-rw hop-type? te-hop-type
          +-:(numbered-link-hop)
            +-rw numbered-link-hop
              +-rw link-tp-id te-tp-id
              +-rw hop-type? te-hop-type
              +-rw direction? te-link-direction
+---:(unnumbered-link-hop)
  |  +--rw unnumbered-link-hop
  |      +--rw link-tp-id
  |      |      te-tp-id
  |  +--rw node-id
  |      te-node-id
  |  +--rw hop-type?
  |      te-hop-type
  |  +--rw direction?
  |      te-link-direction
+---:(as-number)
  |  +--rw as-number-hop
  |      +--rw as-number
  |      |      inet:as-number
  |  +--rw hop-type?
  |      te-hop-type
+---:(label)
  |  +--rw label-hop
  |      +--rw te-label
  |      |      +---:(generic)
  |      |      +--rw generic?
  |      |          rt-types:generalized-label
  |      +--rw direction?
  |          te-label-direction
  +---:(srlg)
    |  +--rw srlg
    |      +--rw srlg?   uint32
    +--rw explicit-route-include-objects
      +--rw route-object-include-object*
          [index]
      |  +--rw index
      |      uint32
      +--rw (type)?
+---:(numbered-node-hop)
  |  +--rw numbered-node-hop
  |      +--rw node-id
  |      |      te-node-id
  |  +--rw hop-type?
  |      te-hop-type
+---:(numbered-link-hop)
  |  +--rw numbered-link-hop
  |      +--rw link-tp-id
  |      |      te-tp-id
  |  +--rw hop-type?
  |      te-hop-type
  |  +--rw direction?
  |      te-link-direction
+--rw upper-bound?  uint64
++--rw path-affinities-values
   +--rw path-affinities-value?  [usage]
      +--rw usage  identityref
      +--rw value?  admin-groups
++--rw path-affinity-names
   +--rw path-affinity-name?  [usage]
      +--rw usage  identityref
      +--rw affinity-name?  [name]
         +--rw name  string
++--rw path-srlgs-lists
   +--rw path-sr1gs-list?  [usage]
      +--rw usage  identityref
      +--rw values?  srlg
++--rw path-srlgs-names
   +--rw path-srlgs-name?  [usage]
      +--rw usage  identityref
      +--rw names?  string
++--rw disjointness?
    +--rw te-path-disjointness
++--rw explicit-route-objects-always
   +--rw route-object-exclude-always?  [index]
      +--rw index  uint32
      +--rw (type)?
         +--:(numbered-node-hop)
            +--rw numbered-node-hop
               +--rw node-id  te-node-id
               +--rw hop-type?  te-hop-type
         +--:(numbered-link-hop)
            +--rw numbered-link-hop
               +--rw link-tp-id  te-tp-id
               +--rw hop-type?  te-hop-type
               +--rw direction?  te-link-direction
         +--:(unnumbered-link-hop)
            +--rw unnumbered-link-hop
               +--rw link-tp-id  te-tp-id
               +--rw node-id  te-node-id
               +--rw hop-type?  te-hop-type
               +--rw direction?  te-link-direction
         +--:(as-number)
            +--rw as-number-hop
               +--rw as-number  inet:as-number
               +--rw hop-type?  te-hop-type
         +--:(label)
            +--rw label-hop
               +--rw te-label
               +--rw (technology)?
                  +--:(generic)
++-rw generic?
   rt-types:generalized-label
++-rw direction?
   te-label-direction
++-rw route-object-include-exclude* [index]
++-rw explicit-route-usage? identityref
++-rw index uint32
++-rw (type)?
   +++-(numbered-node-hop)
      ++-rw numbered-node-hop
      ++-rw node-id te-node-id
      ++-rw hop-type? te-hop-type
   +++-(numbered-link-hop)
      ++-rw numbered-link-hop
      ++-rw link-tp-id te-tp-id
      ++-rw hop-type? te-hop-type
      ++-rw direction? te-link-direction
   +++-(unnumbered-link-hop)
      ++-rw unnumbered-link-hop
      ++-rw link-tp-id te-tp-id
      ++-rw node-id te-node-id
      ++-rw hop-type? te-hop-type
      ++-rw direction? te-link-direction
   +++-(as-number)
      ++-rw as-number-hop
      ++-rw as-number inet:as-number
      ++-rw hop-type? te-hop-type
   +++-(label)
      ++-rw label-hop
      ++-rw te-label
         +++-rw (technology)?
            +++-(generic)
               ++-rw generic?
               rt-types:generalized-label
         ++-rw direction?
            te-label-direction
   +++-(srlg)
      ++-rw srlg
      ++-rw srlg? uint32
++-rw shared-resources-tunnels
   +++-rw lsp-shared-resources-tunnel* tunnel-ref
++-rw path-in-segment!
   +++-rw label-restrictions
      +++-rw label-restriction* [index]
         ++-rw restriction? enumeration
         ++-rw index uint32
      ++-rw label-start
         +++-rw te-label
++-rw (technology)?
   +++-(generic)
      +++-rw generic?
         rt-types:generalized-label
types:generalized-label
   +++-rw direction?
      te-label-direction
++-rw label-end
   +++-rw te-label
      +++-rw (technology)?
         +++-(generic)
            +++-rw generic?
               rt-types:generalized-label
types:generalized-label
      +++-rw direction?
         te-label-direction
++-rw label-step
   +++-rw (technology)?
      +++-(generic)
         +++-rw generic?  int32
      +++-rw range-bitmap?  yang:hex-string
++-rw path-out-segment!
   +++-rw label-restrictions
      +++-rw label-restriction* [index]
         +++-rw restriction?  enumeration
      +++-rw index  uint32
      +++-rw label-start
         +++-rw te-label
            +++-rw (technology)?
               +++-(generic)
                  +++-rw generic?
                     rt-types:generalized-label
types:generalized-label
            +++-rw direction?
               te-label-direction
++-rw label-end
   +++-rw te-label
      +++-rw (technology)?
         +++-(generic)
            +++-rw generic?
               rt-types:generalized-label
types:generalized-label
      +++-rw direction?
         te-label-direction
++-rw label-step
   +++-rw (technology)?
      +++-(generic)
         +++-rw generic?  int32
      +++-rw range-bitmap?  yang:hex-string
++-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    |

| +++-ro computed-paths-properties |
| ++--ro computed-path-properties*[k-index] |
| +++-ro k-index  uint8 |

| +++-ro path-properties |
| ++--ro path-metric*[metric-type] |
| +++-ro metric-type identityref |
| +++-ro accumulative-value? uint64 |

| +++-ro path-affinities-values |
| ++--ro path-affinities-value*[usage] |
| +++-ro usage identityref |
| +++-ro value? admin-groups |

| +++-ro path-affinity-names |
| ++--ro path-affinity-name*[usage] |
| +++-ro usage identityref |
| +++-ro affinity-name*[name] |
| +++-ro name string |

| +++-ro path-srlgs-lists |
| ++--ro path-srlgs-list*[usage] |
| +++-ro usage identityref |
| +++-ro values* srlg |

| +++-ro path-srlgs-names |
| ++--ro path-srlgs-name*[usage] |
| +++-ro usage identityref |
| +++-ro names* string |

| +++-ro path-route-objects |
| ++--ro path-computed-route-object*[index] |
| +++-ro index.uint32 |
| +++-ro (type)? |
| ---:(numbered-node-hop) |
| ++--ro numbered-node-hop |
| +++-ro node-id te-node-id |
| +++-ro hop-type? te-hop-type |
| ---:(numbered-link-hop) |
| ++--ro numbered-link-hop |
| +++-ro link-tp-id te-tp-id |
---ro error-timestamp?
    | yang:date-and-time
---ro error-node-id?
    | te-types:te-node-id
---ro error-link-id?
    | te-types:te-tp-id
---ro source?
    | te-types:te-node-id
---ro destination?
    | te-types:te-node-id
---ro tunnel-id?
    | uint16
---ro lsp-id
    | uint16
---ro extended-tunnel-id?
    | yang:dotted-quad
---ro operational-state?
    | identityref
---ro path-setup-protocol?
    | identityref
---ro origin-type?
    | enumeration
---ro lsp-resource-status?
    | enumeration
---ro lockout-of-normal?
    | boolean
---ro freeze?
    | boolean
---ro lsp-protection-role?
    | enumeration
---ro lsp-protection-state?
    | identityref
---ro protection-group-ingress-node-id?
    | te-types:te-node-id
---ro protection-group-egress-node-id?
    | te-types:te-node-id
---ro lsp-shared-resources-tunnel?
    | tunnel-ref
---ro lsp-record-route-information
    | ---ro lsp-record-route-information* [index]
    | ---ro index uint32
    | ---ro {type}?
    |     +--:(numbered-node-hop)
    |         | ---ro numbered-node-hop
    |         |     | ---ro node-id te-node-id
    |         |     | ---ro flags* path-attribute-flags
    |     +--:(numbered-link-hop)
Internet-Draft   TE YANG Data Model   April 2019

++-ro numbered-node-hop
    +++-ro node-id    te-node-id
    +++-ro hop-type?  te-hop-type
++:-(numbered-link-hop)
++-ro numbered-link-hop
    +++-ro link-tp-id  te-tp-id
    +++-ro hop-type?  te-hop-type
    +++-ro direction?
        te-link-direction
++:-(unnumbered-link-hop)
++-ro unnumbered-link-hop
    +++-ro link-tp-id  te-tp-id
    +++-ro node-id    te-node-id
    +++-ro hop-type?  te-hop-type
    +++-ro direction?
        te-link-direction
++:-(as-number)
++-ro as-number-hop
    +++-ro as-number
        inet:as-number
    +++-ro hop-type?  te-hop-type
++:-(label)
++-ro label-hop
    +++-ro te-label
        +++-ro (technology)?
            +++-:-(generic)
                +++-ro generic?
                    rt-types:generalized-label
                +++-ro direction?
                    te-label-direction
++-ro shared-resources-tunnels
    +++-ro lsp-shared-resources-tunnel*
        tunnel-ref
++-ro te-dev:lsp-timers
    +++-ro te-dev:life-time?  uint32
    +++-ro te-dev:time-to-install?  uint32
    +++-ro te-dev:time-to-destroy?  uint32
++-ro te-dev:downstream-info
    +++-ro te-dev:nhop?
        inet:ip-address
    +++-ro te-dev:outgoing-interface?
        if:interface-ref
    +++-ro te-dev:neighbor?
        inet:ip-address
---x tunnel-action
  +---w input
    |   +---w action-type? identityref
    +---ro output
    +---ro action-result? identityref

---x protection-external-commands
  +---w input
    |   +---w protection-external-command? identityref
    |   +---w protection-group-ingress-node-id? te-types:te-node-id
    |   +---w protection-group-egress-node-id? te-types:te-node-id
    |   +---w path-ref? path-ref
    |   +---w traffic-type? enumeration
    |   +---w extra-traffic-tunnel-ref? tunnel-ref

---rw te-dev:lsp-install-interval? uint32
---rw te-dev:lsp-cleanup-interval? uint32
---rw te-dev:lsp-invalidation-interval? uint32

---rw tunnel-p2mp* [name]
  +---rw name string
  +---rw identifier? uint16
  +---rw description? string
  +---ro operational-state? identityref

---ro lsps-state
  +---ro lsp* [source destination tunnel-id lsp-id extended-tunnel-id]
    +---ro source
      |   te-types:te-node-id
    +---ro destination
      |   te-types:te-node-id
    +---ro tunnel-id uint16
    +---ro lsp-id uint16
    +---ro extended-tunnel-id yang:dotted-quad
    +---ro operational-state? identityref
    +---ro path-setup-protocol? identityref
    +---ro origin-type? enumeration
    +---ro lsp-resource-status? enumeration
    +---ro lockout-of-normal? boolean
    +---ro freeze? boolean
    +---ro lsp-protection-role? enumeration
    +---ro lsp-protection-state? identityref

+--ro protection-group-ingress-node-id?
    te-types:te-node-id
+--ro protection-group-egress-node-id?
    te-types:te-node-id
+--ro lsp-record-route-information
    +--ro lsp-record-route-information* [index]
    +--ro index                        uint32
    +--ro (type)?
        +--:(numbered-node-hop)
            +--ro numbered-node-hop
                +--ro node-id    te-node-id
                +--ro flags*     path-attribute-flags
            +--:(numbered-link-hop)
                +--ro numbered-link-hop
                    +--ro link-tp-id    te-tp-id
                    +--ro flags*        path-attribute-flags
            +--:(unnumbered-link-hop)
                +--ro unnumbered-link-hop
                    +--ro link-tp-id    te-tp-id
                    +--ro node-id?      te-node-id
                    +--ro flags*        path-attribute-flags
            +--:(label)
                +--ro label-hop
                    +--ro te-label
                        +--ro (technology)?
                            +--:(generic)
                                +--ro generic?
                                    rt-types:generalized-label
                                +--ro direction?       te-label-direction
                    +--ro flags*     path-attribute-flags
            +--ro te-dev:lsp-timers
                +--ro te-dev:life-time?         uint32
                +--ro te-dev:time-to-install?   uint32
                +--ro te-dev:time-to-destroy?   uint32
            +--ro te-dev:downstream-info
                +--ro te-dev:nhop?                 inet:ip-address
                +--ro te-dev:outgoing-interface?   if:interface-ref
                +--ro te-dev:neighbor?             inet:ip-address
                +--ro te-dev:label?
                    rt-types:generalized-label
            +--ro te-dev:upstream-info
                +--ro te-dev:phop?                  inet:ip-address
                +--ro te-dev:neighbor?             inet:ip-address
                +--ro te-dev:label?                  rt-types:generalized-label
            +--rw te-dev:interfaces
                +--rw te-dev:threshold-type?        enumeration
                +--rw te-dev:delta-percentage?      rt-types:percentage
                +--rw te-dev:threshold-specification? enumeration
++-rw te-dev:up-thresholds*                          rt-types:percentage
++-rw te-dev:down-thresholds*                        rt-types:percentage
++-rw te-dev:up-down-thresholds*                    rt-types:percentage
++-rw te-dev:interface* [interface]
   ++-rw te-dev:interface
       if:interface-ref
   ++-rw te-dev:te-metric?
       te-types:te-metric
   ++-rw (te-dev:admin-group-type)?
       ++-:(te-dev:value-admin-groups)
           ++-rw (te-dev:value-admin-group-type)?
               ++-:(te-dev:admin-groups)
                   ++-rw te-dev:admin-group?
                       te-types:admin-group
                   ++-:(te-dev:extended-admin-groups)
                       {te-types:extended-admin-groups}?
                       ++-rw te-dev:extended-admin-group?
                       te-types:extended-admin-group
               ++-:(te-dev:named-admin-groups)
               ++-rw te-dev:named-admin-groups* [named-admin-group]
                   ++-rw te-dev:named-admin-group    leafref
       ++-:(te-dev:srlg-type)?
           ++-:(te-dev:value-srlgs)
               ++-rw te-dev:values* [value]
                   ++-rw te-dev:value    uint32
               ++-:(te-dev:named-srlgs)
               ++-rw te-dev:named-srlgs* [named-srlg]
                   {te-types:named-srlg-groups}?
                       ++- rw te-dev:named-srlg    leafref
       ++-rw te-dev:threshold-type?
           enumeration
       ++-rw te-dev:delta-percentage?
           rt-types:percentage
       ++-rw te-dev:threshold-specification?
           enumeration
       ++-rw te-dev:up-thresholds*
           rt-types:percentage
       ++-rw te-dev:down-thresholds*
           rt-types:percentage
       ++-rw te-dev:up-down-thresholds*
           rt-types:percentage
       ++-rw te-dev:switching-capabilities* [switching-capability]
           ++-rw te-dev:switching-capability identityref
           ++-rw te-dev:encoding? identityref
       ++-ro te-dev:state
           ++-ro te-dev:flood-interval?    uint32
           ++-ro te-dev:last-flooded-time? uint32
Internet-Draft             TE YANG Data Model                 April 2019

Figure 3: TE generic model configuration and state tree

3. Model Organization

The TE generic YANG data module "ietf-te" covers configuration, state, RPC and notifications data pertaining to TE global, tunnels and LSPs parameters that are device independent.

The container "te" is the top level container in the data model. The presence of this container enables TE function system wide.

The model top level organization is shown below in Figure 4:
module: ietf-te
  +--rw te!
    +--rw globals
      .
      .
    +--rw tunnels
      .
      .
    --- lsps-state

rpcs:
  ----x globals-rpc
  ----x tunnels-rpc
notifications:
  ----n globals-notif
  ----n tunnels-notif

Figure 4: TE generic highlevel model view

3.1. Global Configuration and State Data

The global TE branch of the data model covers configurations that control TE features behavior system-wide, and its respective state. Examples of such configuration data are:

- Table of named SRLG mappings
- Table of named (extended) administrative groups mappings
- Table of named path-constraints sets
- System-wide capabilities for LSP reoptimization
  * Reoptimization timers (periodic interval, LSP installation and cleanup)
  * Link state flooding thresholds
  * Periodic flooding interval
- Global capabilities that affect originating, transiting and terminating LSPs. For example:
  * Path selection parameters (e.g. metric to optimize, etc.)
  * Path or segment protection parameters
3.2. Interfaces Configuration and State Data

This branch of the model covers configuration and state data corresponding to TE interfaces that are present on a device. The module "ietf-te-device" is introduced to hold such TE device specific properties.

Examples of TE interface properties are: * Maximum reservable bandwidth, bandwidth constraints (BC) * Flooding parameters * Flooding intervals and threshold values * Interface attributes * (Extended) administrative groups * SRLG values * TE metric value * Fast reroute backup tunnel properties (such as static, auto-tunnel)

module: ietf-te-device
  augment /te:te:
    +--rw interfaces
      .
      +-- rw te-dev:te-attributes
        <<intended configuration>>
          .
          +-- ro state
            <<derived state associated with the TE interface>>

Figure 5: TE interface state

The derived state associated with interfaces is grouped under the interface "state" sub-container as shown in Figure 5. This covers state data such as:

- Bandwidth information: maximum bandwidth, available bandwidth at different priorities and for each class-type (CT)
- List of admitted LSPs
  * Name, bandwidth value and pool, time, priority
- Statistics: state counters, flooding counters, admission counters (accepted/rejected), preemption counters
- Adjacency information
  * Neighbor address
  * Metric value
3.3. Tunnels Configuration and State Data

This branch covers data related to TE tunnels configuration and state. The derived state associated with tunnels is grouped under a state container as shown in Figure 6.

```
module: ietf-te
    +--rw te!
        +--rw tunnels
            <<intended configuration>>
        +-- ro state
            <<derived state associated with the tunnel>>
```

Figure 6: TE interface state tree

Examples of tunnel configuration data for TE tunnels:

- Name and type (e.g. P2P, P2MP) of the TE tunnel
- Administrative and operational state of the TE tunnel
- Set of primary and corresponding secondary paths and corresponding path attributes
- Bidirectional path attribute(s) including forwarding and reverse path properties
- Protection and restoration path parameters

3.3.1. Tunnel Compute-Only Mode

A configured TE tunnel, by default, is provisioned so it can carry traffic as soon as a valid path is computed and an LSP instantiated. In some cases, however, a TE tunnel may be provisioned for the only purpose of computing a path and reporting it without the need to instantiate the LSP or commit any resources. In such a case, the tunnel is configured in "compute-only" mode to distinguish it from default tunnel behavior.

A "compute-only" TE tunnel is configured as a usual TE tunnel with associated per path constraint(s) and properties on a device or controller. The device or controller computes the feasible path(s) subject to configured constraints and reflects the computed path(s) in the LSP(s) Record-Route Object (RRO) list. At any time, a client may query "on-demand" the "compute-only" TE tunnel computed path(s) properties by querying the state of the tunnel. Alternatively, the
client can subscribe on the "compute-only" TE tunnel to be notified of computed path(s) and whenever it changes.

3.3.2. Tunnel Hierarchical Link Endpoint

TE LSPs can be set up in MPLS or Generalized MPLS (GMPLS) networks to be used to form links to carry traffic in other (client) networks [RFC6107]. In this case, the model introduces the TE tunnel hierarchical link endpoint parameters to identify the specific link in the client layer that the underlying TE tunnel is associated with.

3.4. TE LSPs State Data

TE LSPs are derived state data that are present whenever the LSP(s) are instantiated - for example, when associated signaling completes. TE LSPs exists on routers as ingress (starting point of LSP), transit (mid-point of LSP), or egress (termination point of the LSP). In the model, the nodes holding TE LSP data exist in the read-only lsps-state list as shown in Figure 3.

3.5. Global RPC Data

This branch of the model covers system-wide RPC execution data to trigger actions and optionally expect responses. Examples of such TE commands are to:

- Clear global TE statistics of various features

3.6. Interface RPC Data

This collection of data in the model defines TE interface RPC execution commands. Examples of these are to:

- Clear TE statistics for all or for individual TE interfaces
- Trigger immediate flooding for one or all TE interfaces

3.7. Tunnel RPC Data

This branch of the model covers TE tunnel RPC execution data to trigger actions and expect responses. The TE generic YANG data model defines target containers that an external module in [I-D.ietf-teas-yang-path-computation] augments with RPCs that allow the invocation of certain TE functions (e.g. path computations).
4. TE Generic and Helper YANG Modules

The TE generic YANG module "ietf-te" imports the following modules:

- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]

This module references the following documents: [RFC6991], [RFC4875], [RFC7551], [RFC4206], [RFC4427], [RFC4872], [RFC3945], [RFC3209], [RFC4872], [RFC6780], and [RFC7308].

<CODE BEGINS> file "ietf-te@2019-04-09.yang"
module ietf-te {
    yang-version 1.1;

    namespace "urn:ietf:params:xml:ns:yang:ietf-te";

    /* Replace with IANA when assigned */
    prefix "te";

    /* Import TE generic types */
    import ietf-te-types {
        prefix te-types;
        reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering Types";
    }

    import ietf-inet-types {
        prefix inet;
        reference "RFC6991: Common YANG Data Types";
    }

    import ietf-yang-types {
        prefix "yang";
        reference "RFC6991: Common YANG Data Types";
    }

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

    contact
    "WG Web:  <http://tools.ietf.org/wg/teas/>
    WG List:  <mailto:teas@ietf.org>
    WG Chair: Lou Berger
                <mailto:lberger@labn.net>"
description
"YANG data module for TE configuration, state, RPC and notifications.";

revision "2019-04-09" {
  description "Latest update to TE generic YANG module.";
  reference
  "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

identity path-computation-error-reason {
  description
  "Base identity for path computation error reasons";
}

identity path-computation-error-no-topology {
  base path-computation-error-reason;
  description
  "Path computation error no topology error reason";
}

identity path-computation-error-no-server {
  base path-computation-error-reason;
  description
  "Path computation error no server error reason";
}

identity path-computation-error-path-not-found {
  base path-computation-error-reason;
typedef tunnel-ref {
  type leafref {
    path "te:tunnel/te:name";
  }
  description
    "This type is used by data models that need to reference configured TE tunnel.";
}

typedef tunnel-p2mp-ref {
  type leafref {
    path "te:tunnel-p2mp/te:name";
  }
  description
    "This type is used by data models that need to reference configured P2MP TE tunnel.";
  reference "RFC4875";
}

typedef path-ref {
  type union {
    type leafref {
      path "te:tunnel/te:p2p-primary-paths/te:p2p-primary-path/te:name";
    }
    type leafref {
      path "te:tunnel/te:p2p-secondary-paths/te:p2p-secondary-path/te:name";
    }
  }
  description
    "This type is used by data models that need to reference configured primary or secondary path of a TE tunnel.";
}

/**
 * TE tunnel generic groupings
 */
grouping p2p-secondary-path-properties {
  description "tunnel path properties.";
  uses p2p-path-properties;
  uses path-constraints-common;
  uses protection-restoration-properties;
  uses p2p-path-properties-state;
grouping p2p-primary-path-properties {
  description "TE tunnel primary path properties grouping";
  uses p2p-path-properties;
  uses path-constraints-common;
  uses p2p-path-properties-state;
}

grouping path-properties {
  description "TE computed path properties grouping";
  container path-properties {
    description "The TE path computed properties";
    list path-metric {
      key metric-type;
      description "TE path metric type";
      leaf metric-type {
        type identityref {
          base te-types:path-metric-type;
        }
        description "TE path metric type";
      }
      leaf accumulative-value {
        type uint64;
        description "TE path metric accumulative value";
      }
    }
  }
}

uses te-types:generic-path-affinities;
uses te-types:generic-path-srlgs;
container path-route-objects {
  config 'false';
  description "Container for the list of computed route objects as returned by the computation engine";
  list path-computed-route-object {
    key index;
    ordered-by user;
    description "List of computed route objects returned by the computation engine";
    leaf index {
      type uint32;
      description "Route object entry index. The index is used to identify an entry in the list. The order of entries is defined by the user without relying on key values";
    }
  }
}
uses te-types:explicit-route-hop;
}
}
uses shared-resources-tunnels;
}
}

grouping p2p-path-properties-state {
  description "TE per path state parameters";
  container computed-paths-properties {
    config 'false';
    description "Computed path properties container";
    list computed-path-properties {
      key k-index;
      description "List of computed paths";
      leaf k-index {
        type uint8;
        description "The k-th path returned from the computation server. A lower k value path is more optimal than higher k value path(s)";
      }
      uses path-properties {
        description "The TE path computed properties";
      }
    }
    uses computed-path-error-info;
    uses lsp-provisioning-error-info {
      augment "lsp-provisioning-error-infos/" + "lsp-provisioning-error-info" {
        description "Augmentation of LSP provisioning information under a specific path";
        leaf lsp-id {
          type uint16;
          description "The LSP-ID for which path compatutation was performed.";
        }
      }
    }
  }
}

container lsps {
  config 'false';
  description "TE LSPs container";
  list lsp {
    key "lsp-id";
    description "List of LSPs associated with the tunnel.";
  }
uses lsp-provisioning-error-info;
uses lsp-properties-state;
uses shared-resources-tunnels-state;
uses lsp-record-route-information-state;
uses path-properties {
    description "The TE path actual properties";
}

}
}
grouping computed-path-error-info {
    description "Grouping for path computation error information";
    container computed-path-error-infos {
        config false;
        description "Path computation information container";
        list computed-path-error-info {
            description "List of path computation info entries";
            leaf error-description {
                type string;
                description "Textual representation of the error occurred during path computation.";
            }
            leaf error-timestamp {
                type yang:date-and-time;
                description "Timestamp of last path computation attempt.";
            }
            leaf error-reason {
                type identityref {
                    base path-computation-error-reason;
                }
                description "Reason for the path computation error.";
            }
        }
    }
}
grouping lsp-provisioning-error-info {
    description "Grouping for LSP provisioning error information";
    container lsp-provisioning-error-infos {
        config false;
    }
}
description
"LSP provisioning error information";
list lsp-provisioning-error-info {
    description
    "List of LSP provisioning error info entries";
    leaf error-description {
        type string;
        description
        "Textual representation of the error occurred during path computation.";
    }
    leaf error-timestamp {
        type yang:date-and-time;
        description
        "Timestamp of when the reported error occurred.";
    }
    leaf error-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
        "Node identifier of node where error occurred.";
    }
    leaf error-link-id {
        type te-types:te-tp-id;
        default 0;
        description
        "Link ID where the error occurred.";
    }
}

grouping p2p-path-properties-common {
    description
    "TE tunnel common path properties configuration grouping";
    leaf name {
        type string;
        description "TE path name";
    }
    leaf path-setup-protocol {
        type identityref {
            base te-types:path-signaling-type;
        }
        default te-types:path-setup-static;
        description
        "Signaling protocol used to set up this tunnel";
    }
    leaf path-computation-method {
        type identityref {
            base te-types:path-computation-type;
        }
        default te-types:path-computation-static;
        description
        "Path computation method used to compute this path";
    }
}

type identityref {
    base te-types:path-computation-method;
}
default te-types:path-locally-computed;
description
    "The method used for computing the path, either
    locally computed, queried from a server or not
    computed at all (explicitly configured).";
}
leaf path-computation-server {
    when "../path-computation-method = "+ "te-types:path-externally-queried" {
        description
            "The path-computation server when the path is
            externally queried";
    }
    type inet:ip-address;
description
    "Address of the external path computation
    server";
}
leaf compute-only {
    type empty;
description
    "When set, the path is computed and updated whenever
    the topology is updated. No resources are committed
    or reserved in the network.";
}
leaf use-path-computation {
    when "../path-computation-method = " + "te-types:path-locally-computed";
    type boolean;
default 'true';
description "A CSPF dynamically computed path";
}
leaf lockdown {
    type empty;
description
    "Indicates no reoptimization to be attempted for
    this path.";
}
leaf path-scope {
    type identityref {
        base te-types:path-scope-type;
    }
    default te-types:path-scope-end-to-end;
    config 'false';
description "Path scope if segment or an end-to-end path";
grouping p2p-reverse-path-properties {
  description "TE tunnel reverse path properties configuration grouping";
  uses p2p-path-properties-common;
  uses te-types:generic-path-optimization;
  leaf named-path-constraint {
    if-feature te-types:named-path-constraints;
    type leafref {
      path "../../../../../../globals/"
      + "named-path-constraints/named-path-constraint/"
      + "name";
    }
    description "Reference to a globally defined named path constraint set";
  }
}

grouping p2p-primary-reverse-path-properties {
  description "TE P2P tunnel primary reverse path properties.";
  reference "RFC7551";
  container p2p-primary-reverse-path {
    description "Tunnel reverse primary path properties";
    uses p2p-reverse-path-properties;
    uses path-constraints-common;
    uses p2p-path-properties-state;
    container p2p-secondary-reverse-path {
      description "Tunnel reverse secondary path properties";
      uses p2p-secondary-reverse-path-properties;
    }
  }
}

grouping p2p-path-properties {
  description "TE tunnel path properties configuration grouping";
  uses p2p-path-properties-common;
  uses te-types:generic-path-optimization;
  leaf preference {
    type uint8 { 
      range "1..255";
    }
    default 1;
    description

" Specifies a preference for this path. The lower the number higher the preference " ;

} leaf k-requested-paths {
  type uint8;
  default 1;
  description
  " The number of k-shortest-paths requested from the path computation server and returned sorted by its optimization objective " ;
}

leaf named-path-constraint {
  if-feature te-types:named-path-constraints;
  type leafref {
    path "../../../globals/" + "named-path-constraints/named-path-constraint/" + "name";
  }
  description
  " Reference to a globally defined named path constraint set " ;
}

} grouping hierarchical-link-properties {
  description
  " Hierarchical link grouping " ;
  reference " RFC4206 " ;
  container hierarchical-link {
    description
    " Identifies a hierarchical link (in client layer) that this tunnel is associated with." ;
    leaf local-te-node-id {
      type te-types:te-node-id;
      default "0.0.0.0";
      description
      " Local TE node identifier " ;
    }
    leaf local-te-link-tp-id {
      type te-types:te-tp-id;
      default 0;
      description
      " Local TE link termination point identifier " ;
    }
    leaf remote-te-node-id {
      type te-types:te-node-id;
      default "0.0.0.0";
      description

"Remote TE node identifier";
}
uses te-types:te-topology-identifier;
}

grouping protection-restoration-properties-state {
    description "Protection parameters grouping";
    leaf lockout-of-normal {
        type boolean;
        default 'false';
        description "When set to 'True', it represents a lockout of normal traffic external command. When set to 'False', it represents a clear lockout of normal traffic external command. The lockout of normal traffic command applies to this Tunnel."
    }
    reference "RFC4427";
}
leaf freeze {
    type boolean;
    default 'false';
    description "When set to 'True', it represents a freeze external command. When set to 'False', it represents a clear freeze external command. The freeze command command applies to all the Tunnels which are sharing the protection resources with this Tunnel."
    reference "RFC4427";
}
leaf lsp-protection-role {
    type enumeration {
    enum working {
        description "A working LSP must be a primary LSP whilst a protecting LSP can be either a primary or a secondary LSP. Also, known as protected LSPs when working LSPs are associated with protecting LSPs.";
    }
    enum protecting {
        description "A secondary LSP is an LSP that has been provisioned in the control plane only; e.g. resource allocation
leaf lsp-protection-state {
    type identityref {
        base te-types:lsp-protection-state;
    }
    default te-types:normal;
    description "The state of the APS state machine controlling which
tunnels is using the resources of the protecting LSP."
}

leaf protection-group-ingress-node-id {
    type te-types:te-node-id;
    default "0.0.0.0";
    description "Indicates the te-node-id of the protection group
    ingress node when the APS state represents an extenal
    command (LoP, SF, MS) applied to it or a WTR timer
    running on it. If the external command is not applied to
    the ingress node or the WTR timer is not running on it,
    this attribute is not specified. A value 0.0.0.0 is used
    when the te-node-id of the protection group ingress node is
    unknown (e.g., because the ingress node is outside the scope
    of control of the server)"
}

leaf protection-group-egress-node-id {
    type te-types:te-node-id;
    default "0.0.0.0";
    description "Indicates the te-node-id of the protection group egress node
    when the APS state represents an extenal command (LoP, SF, MS) applied to it or a WTR timer running on it. If the external command is not applied to the ingress node or the WTR timer is not running on it, this attribute is not specified. A value 0.0.0.0 is used when the te-node-id of the protection group ingress node is unknown (e.g., because the ingress node is outside the scope of control of the server)"
}
description "Protection and restoration parameters";
container protection {
  description "Protection parameters";
  leaf enable {
    type boolean;
    default 'false';
    description "A flag to specify if LSP protection is enabled";
    reference "RFC4427";
  }
  leaf protection-type {
    type identityref {
      base te-types:lsp-protection-type;
    }
    default te-types:lsp-protection-unprotected;
    description "LSP protection type.";
  }
  leaf protection-reversion-disable {
    type boolean;
    default 'false';
    description "Disable protection reversion to working path";
  }
  leaf hold-off-time {
    type uint32;
    units "milli-seconds";
    default 0;
    description "The time between the declaration of an SF or SD condition and the initialization of the protection switching algorithm.";
    reference "RFC4427";
  }
  leaf wait-to-revert {
    type uint16;
    units seconds;
    description "Time to wait before attempting LSP reversion";
    reference "RFC4427";
  }
  leaf aps-signal-id {
    type uint8 {
      range "1..255";
    }
    default 1;
    description "The APS signal number used to reference the traffic of this tunnel. The default value for normal traffic is 1. The default value for extra-traffic is 255. If not specified,
  "}
non-default values can be assigned by the server, if and only if, the server controls both endpoints.
reference "RFC4427";
}
}
container restoration {
    description "Restoration parameters";
    leaf enable {
        type boolean;
        default 'false';
        description "A flag to specify if LSP restoration is enabled";
        reference "RFC4427";
    }
    leaf restoration-type {
        type identityref {
            base te-types:lsp-restoration-type;
        }  
        default te-types:lsp-restoration-restore-any;
        description "LSP restoration type.";
    }
    leaf restoration-scheme {
        type identityref {
            base te-types:restoration-scheme-type;
        }
        default te-types:restoration-scheme-preconfigured;
        description "LSP restoration scheme.";
    }
    leaf restoration-reversion-disable {
        type boolean;
        default 'false';
        description "Disable restoration reversion to working path";
    }
    leaf hold-off-time {
        type uint32;
        units "milli-seconds";
        description "The time between the declaration of an SF or SD condition and the initialization of the protection switching algorithm.";
        reference "RFC4427";
    }
    leaf wait-to-restore {
        type uint16;
        units seconds;
        description "Time to wait before attempting LSP restoration";
        reference "RFC4427";
    }
leaf wait-to-revert {
    type uint16;
    units seconds;
    description "Time to wait before attempting LSP reversion";
    reference "RFC4427";
}

grouping p2p-dependency-tunnels-properties {
    description "Grouping for tunnel dependency list of tunnels";
    container dependency-tunnels {
        description "Dependency tunnels list";
        list dependency-tunnel {
            key "name";
            description "Dependency tunnel entry";
            leaf name {
                type leafref {
                    path "../../../tunnels/tunnel/name";
                    require-instance 'false';
                }
                description "Dependency tunnel name";
            }
            leaf encoding {
                type identityref {
                    base te-types:lsp-encoding-types;
                }
                default te-types:lsp-encoding-packet;
                description "LSP encoding type";
                reference "RFC3945";
            }
            leaf switching-type {
                type identityref {
                    base te-types:switching-capabilities;
                }
                default te-types:switching-psc1;
                description "LSP switching type";
                reference "RFC3945";
            }
        }
    }
}

grouping tunnel-p2p-config {
    description

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

leaf identifier {
    type uint16;
    description "TE tunnel Identifier.";
    reference "RFC3209";
}

leaf description {
    type string;
    default 'None';
    description "Textual description for this TE tunnel";
}

leaf encoding {
    type identityref {
        base te-types: lsp-encoding-types;
    }
    default te-types: lsp-encoding-packet;
    description "LSP encoding type";
    reference "RFC3945";
}

leaf switching-type {
    type identityref {
        base te-types: switching-capabilities;
    }
    default te-types: switching-psc1;
    description "LSP switching type";
    reference "RFC3945";
}

leaf provisioning-state {
    type identityref {
        base te-types: tunnel-state-type;
    }
    default te-types: tunnel-state-up;
    description "TE tunnel administrative state.";
}

leaf preference {
    type uint8 {
        range "1..255";
    }
    default 100;
    description "Specifies a preference for this tunnel.
        A lower number signifies a better preference";
}
leaf reoptimize-timer {
    type uint16;
    units seconds;
    description "frequency of reoptimization of a traffic engineered LSP";
}

leaf source {
    type te-types:te-node-id;
    description "TE tunnel source node ID.";
}

leaf destination {
    type te-types:te-node-id;
    description "TE tunnel destination node ID";
}

leaf src-tp-id {
    type yang:hex-string;
    default '00:00:00:00';
    description "TE tunnel source termination point identifier.";
}

leaf dst-tp-id {
    type yang:hex-string;
    default '00:00:00:00';
    description "TE tunnel destination termination point identifier.";
}

leaf bidirectional {
    type boolean;
    default 'false';
    description "TE tunnel bidirectional";
}

uses tunnel-p2p-associations-properties;
uses protection-restoration-properties;
uses te-types:tunnel-constraints;
uses p2p-dependency-tunnels-properties;
uses hierarchical-link-properties;

grouping tunnel-p2p-associations-properties {
    description "TE tunnel association grouping";
    container association-objects {
        description "TE tunnel associations";
        list association-object {
            key "type ID source global-source";
            description "List of association base objects";
            reference "RFC4872";
            leaf type {
                
type identityref {
    base te-types:association-type;
}  
description "Association type";
reference "RFC4872";
}

leaf ID {
    type uint16;
    description "Association ID";
    reference "RFC4872";
}

leaf source {
    type te-types:te-node-id;
    description "Association source";
    reference "RFC4872";
}

leaf global-source {
    type te-types:te-node-id;
    description "Association global source";
    reference "RFC4872";
}

list association-object-extended {
    key "type ID source global-source extended-ID";
    description "List of extended association objects";
    reference "RFC6780";
    leaf type {
        type identityref {
            base te-types:association-type;
        }  
        description "Association type";
    }
    leaf ID {
        type uint16;
        description "Association ID";
        reference "RFC4872";
    }
    leaf source {
        type te-types:te-node-id;
        description "Association source";
    }
    leaf global-source {
        type te-types:te-node-id;
        description "Association global source";
        reference "RFC4872";
    }
    leaf extended-ID {
        type yang:hex-string;
    }
}

Saad, et al. Expires October 11, 2019 [Page 70]
grouping path-access-segment-info {
  description "If an end-to-end tunnel crosses multiple domains using the same technology, some additional constraints have to be taken in consideration in each domain";
  container path-in-segment {
    presence "The end-to-end tunnel starts in a previous domain; this tunnel is a segment in the current domain.";
    description "This tunnel is a segment that needs to be coordinated with previous segment stitched on head-end side.";
    uses te-types:label-set-info;
  }
  container path-out-segment {
    presence "The end-to-end tunnel is not terminated in this domain; this tunnel is a segment in the current domain.";
    description "This tunnel is a segment that needs to be coordinated with previous segment stitched on head-end side.";
    uses te-types:label-set-info;
  }
}

/* TE tunnel configuration/state grouping */
grouping tunnel-p2mp-properties {
  description "Top level grouping for P2MP tunnel properties.";
  leaf name {
    type string;
    description "TE tunnel name.";
  }
  leaf identifier {
    type uint16;
    description "TE tunnel Identifier.";
    reference "RFC3209";
  }
  leaf description {
    type string;
  }
}
default 'None';
description
  "Textual description for this TE tunnel";
}
leaf operational-state {
  type identityref {
    base te-types:tunnel-state-type;
  }
  default te-types:tunnel-state-up;
  config 'false';
  description "TE tunnel administrative state.";
}
}
grouping p2p-path-candidate-secondary-path-config {
  description
    "Configuration parameters relating to a secondary path which is a candidate for a particular primary path";

leaf secondary-path {
  type leafref {
    path "../../../p2p-secondary-paths/" + "p2p-secondary-path/name";
  }
  description
    "A reference to the secondary path that should be utilised when the containing primary path option is in use";
}

leaf path-setup-protocol {
  type identityref {
    base te-types:path-signaling-type;
  }
  default te-types:path-setup-static;
  description
    "Signaling protocol used to set up this tunnel";
}
}
grouping p2p-secondary-reverse-path-properties {
  description
    "Configuration parameters relating to a secondary path which is a candidate for a particular primary path";

leaf secondary-path {
  type leafref {
    path "../../../p2p-secondary-paths/" + "p2p-secondary-path/name";
  }
  description
    "A reference to the secondary path that should be utilised when the containing primary path option is in use";
}
leaf path-setup-protocol {
  type identityref {
    base te-types:path-signaling-type;
  }
  default te-types:path-setup-static;
  description
  "Signaling protocol used to set up this tunnel";
}

grouping tunnel-p2p-properties {
  description
  "Top level grouping for tunnel properties.";
  leaf operational-state {
    type identityref {
      base te-types:tunnel-state-type;
    }
    default te-types:tunnel-state-up;
    config 'false';
    description "TE tunnel administrative state.";
  }
  uses tunnel-p2p-config;
  container p2p-primary-paths {
    description "Set of P2P primary aths container";
    list p2p-primary-path {
      key "name";
      description
      "List of primary paths for this tunnel.";
      uses p2p-primary-path-properties;
      uses p2p-primary-reverse-path-properties;
      container candidate-p2p-secondary-paths {
        description
        "The set of candidate secondary paths which may be used
        for this primary path. When secondary paths are specified
        in the list the path of the secondary LSP in use must be
        restricted to those path options referenced. The
        priority of the secondary paths is specified within the
        list. Higher priority values are less preferred - that is
        to say that a path with priority 0 is the most preferred
        path. In the case that the list is empty, any secondary
        path option may be utilised when the current primary path
        is in use.";
      }
    }
  }
}
list candidate-p2p-secondary-path {
  key "secondary-path";
  description
    "List of secondary paths for this tunnel.";
  uses p2p-path-candidate-secondary-path-config;

  leaf active {
    type boolean;
    config 'false';
    description
      "Indicates the current active path option that has
       been selected of the candidate secondary paths";
  }
}
}

container p2p-secondary-paths {
  description "Set of P2P secondary paths container";
  list p2p-secondary-path {
    key "name";
    description
      "List of secondary paths for this tunnel.";
    uses p2p-secondary-path-properties;
  }
}
}

grouping shared-resources-tunnels-state {
  description
    "The specific tunnel that is using the shared secondary path
     resources";
  leaf lsp-shared-resources-tunnel {
    type tunnel-ref;
    description
      "Reference to the tunnel that sharing secondary path
       resources with this tunnel";
  }
}
}

grouping shared-resources-tunnels {
  description
    "Set of tunnels that share secondary path resources with
     this tunnel";
  container shared-resources-tunnels {
    description
      "Set of tunnels that share secondary path resources with
       this tunnel";
    leaf-list lsp-shared-resources-tunnel {

type tunnel-ref;
description
  "Reference to the tunnel that sharing secondary path
  resources with this tunnel";
}
}

grouping tunnel-actions {
  description "Tunnel actions";
  action tunnel-action {
    description "Tunnel action";
    input {
      leaf action-type {
        type identityref {
          base te-types:tunnel-action-type;
        }
        description "Tunnel action type";
      }
    }
    output {
      leaf action-result {
        type identityref {
          base te-types:te-action-result;
        }
        description "The result of the RPC operation";
      }
    }
  }
}

grouping tunnel-protection-actions {
  description
    "Protection external command actions";
  action protection-external-commands {
    input {
      leaf protection-external-command {
        type identityref {
          base te-types:protection-external-commands;
        }
        description
          "Protection external command";
      }
      leaf protection-group-ingress-node-id {
        type te-types:te-node-id;
        description
          "When specified, indicates whether the action is
          applied on ingress node.
          By default, if neither ingress nor egress node-id
is set, the action applies to ingress node only.
}
leaf protection-group-egress-node-id {
  type te-types:te-node-id;
  description
    "When specified, indicates whether the action is
    applied on egress node.
    By default, if neither ingress nor egress node-id
    is set, the action applies to ingress node only.";
}
leaf path-ref {
  type path-ref;
  description
    "Indicates to which path the external command applies to.";
}
leaf traffic-type {
  type enumeration {
    enum normal-traffic {
      description
        "The manual-switch or forced-switch command applies to
        the normal traffic (this Tunnel).";
    }
    enum null-traffic {
      description
        "The manual-switch or forced-switch command applies to
        the null traffic.";
    }
    enum extra-traffic {
      description
        "The manual-switch or forced-switch command applies to
        the extra traffic (the extra-traffic Tunnel sharing
        protection bandwidth with this Tunnel).";
    }
  }
  description
    "Indicates whether the manual-switch or forced-switch
    commands applies to the normal traffic, the null traffic
    or the extra-traffic.";
  reference "RFC4427";
}
leaf extra-traffic-tunnel-ref {
  type tunnel-ref;
  description
    "In case there are multiple extra-traffic tunnels sharing
    protection bandwidth with this Tunnel (m:n protection),
    represents which extra-traffic Tunnel the manual-switch or
    forced-switch to extra-traffic command applies to.";
}
grouping lsp-record-route-information-state {
    description "recorded route information grouping";
    container lsp-record-route-information {
        description "RSVP recorded route object information";
        list lsp-record-route-information {
            when "../..//origin-type = 'ingress'" {
                description "Applicable on ingress LSPs only";
            }
            key "index";
            description "Record route list entry";
            uses te-types:record-route-state;
        }
    }
}

grouping lsps-state-grouping {
    description "LSPs state operational data grouping";
    container lsps-state {
        config 'false';
        description "TE LSPs state container";
        list lsp {
            key "source destination tunnel-id lsp-id "+ "extended-tunnel-id";
            description "List of LSPs associated with the tunnel.";
            uses lsp-properties-state;
            uses lsp-record-route-information-state;
        }
    }
}

/*** End of TE LSP groupings ***/

/**
 * TE global generic groupings
 */

/* Global named admin-groups configuration data */
grouping named-admin-groups-properties {
  description
    "Global named administrative groups configuration grouping";
  leaf name {
    type string;
    description
      "A string name that uniquely identifies a TE interface named admin-group";
  }
  leaf bit-position {
    type uint32;
    description
      "Bit position representing the administrative group";
    reference "RFC3209 and RFC7308";
  }
}

/* Global named admin-srlgs configuration data */
grouping named-srlgs-properties {
  description
    "Global named SRLGs configuration grouping";
  leaf name {
    type string;
    description
      "A string name that uniquely identifies a TE interface named srlg";
  }
  leaf group {
    type te-types:srlg;
    description "An SRLG value";
  }
}
leaf cost {
    type uint32;
    description
        "SRLG associated cost. Used during path to append
         the path cost when traversing a link with this SRLG";
} }

grouping named-srlgs {
    description
        "Global named SRLGs configuration grouping";
    container named-srlgs {
        description "TE named SRLGs container";
        list named-srlg {
            if-feature te-types:named-srlg-groups;
            key "name";
            description
                "A list of named SRLG groups";
            uses named-srlgs-properties;
        }
    }
}

/* Global named paths constraints configuration data */
grouping path-constraints-state {
    description "TE path constraints state";
    leaf bandwidth {
        type te-types:te-bandwidth;
        config 'false';
        description
            "A technology agnostic requested bandwidth to use
             for path computation";
    }
    leaf disjointness-type {
        type te-types:te-path-disjointness;
        config 'false';
        description
            "The type of resource disjointness.";
    }
}

grouping path-constraints-common {
    description
        "Global named path constraints configuration
grouping";
    uses te-types:common-path-constraints-attributes;
    uses te-types:generic-path-disjointness;
    uses te-types:path-constraints-route-objects;
uses shared-resources-tunnels {
  description
  "Set of tunnels that are allowed to share secondary path
  resources of this tunnel";
}

uses path-access-segment-info {
  description
  "Tunnel constraints induced by other segments.";
}

grouping named-path-constraints {
  description
  "Global named path constraints configuration grouping";
  container named-path-constraints {
    description "TE named path constraints container";
    list named-path-constraint {
      if-feature te-types:named-path-constraints;
      key "name";
      leaf name {
        type string;
        description
        "A string name that uniquely identifies a
        path constraint set";
      }
      uses path-constraints-common;
      description
      "A list of named path constraints";
    }
  }
}

/* TE globals container data */
grouping globals-grouping {
  description
  "Globals TE system-wide configuration data grouping";
  container globals {
    description
    "Globals TE system-wide configuration data container";
    uses named-admin-groups;
    uses named-srlgs;
    uses named-path-constraints;
  }
}

/* TE tunnels container data */
grouping tunnels-grouping {


description
    "Tunnels TE configuration data grouping";
container tunnels {
    description
        "Tunnels TE configuration data container";

list tunnel {
    key "name";
    description "P2P TE tunnels list.";
    uses tunnel-p2p-properties;
    uses tunnel-actions;
    uses tunnel-protection-actions;
}
list tunnel-p2mp {
    key "name";
    unique "identifier";
    description "P2MP TE tunnels list.";
    uses tunnel-p2mp-properties;
}

/* TE LSPs ephemeral state container data */
grouping lsp-properties-state {
    description
        "LSPs state operational data grouping";
    leaf source {
        type te-types:te-node-id;
        description
            "Tunnel sender address extracted from
            SENDER TEMPLATE object";
        reference "RFC3209";
    }
    leaf destination {
        type te-types:te-node-id;
        description
            "Tunnel endpoint address extracted from
            SESSION object";
        reference "RFC3209";
    }
    leaf tunnel-id {
        type uint16;
        description
            "Tunnel identifier used in the SESSION
            that remains constant over the life
            of the tunnel.";
        reference "RFC3209";
    }
}
leaf lsp-id {
    type uint16;
    description
        "Identifier used in the SENDER_TEMPLATE
        and the FILTER_SPEC that can be changed
        to allow a sender to share resources with
        itself.";
    reference "RFC3209";
}
leaf extended-tunnel-id {
    type yang:dotted-quad;
    description
        "Extended Tunnel ID of the LSP.";
    reference "RFC3209";
}
leaf operational-state {
    type identityref {
        base te-types:lsp-state-type;
    }
    description "LSP operational state.";
}
leaf path-setup-protocol {
    type identityref {
        base te-types:path-signaling-type;
    }
    default te-types:path-setup-static;
    description
        "Signaling protocol used to set up this tunnel";
}
leaf origin-type {
    type enumeration {
        enum ingress {
            description
                "Origin ingress";
        }
        enum egress {
            description
                "Origin egress";
        }
        enum transit {
            description
                "transit";
        }
    }
    default 'ingress';
    description
        "Origin type of LSP relative to the location
        of the local switch in the path.";
leaf lsp-resource-status {
    type enumeration {
        enum primary {
            description
            "A primary LSP is a fully established LSP for which the resource allocation has been committed at the data plane";
        }
        enum secondary {
            description
            "A secondary LSP is an LSP that has been provisioned in the control plane only; e.g. resource allocation has not been committed at the data plane";
        }
    }
    default 'primary';
    description "LSP resource allocation type";
    reference "RFC4872, section 4.2.1";
}

uses protection-restoration-properties-state;

/*** End of TE global groupings ***/

/**
 * TE configurations container
 */
container te {
    presence "Enable TE feature.";
    description "TE global container."

    /* TE Global Configuration Data */
    uses globals-grouping;

    /* TE Tunnel Configuration Data */
    uses tunnels-grouping;

    /* TE LSPs State Data */
    uses lsps-state-grouping;

} /* TE Global RPCs/execution Data */
rpc globals-rpc {
    description
"Execution data for TE global.";
}

/* TE interfaces RPCs/execution Data */
rpc interfaces-rpc {
  description
  "Execution data for TE interfaces.";
}

/* TE Tunnel RPCs/execution Data */
rpc tunnels-rpc {
  description "TE tunnels RPC nodes";
  input {
    container tunnel-info {
      description "Tunnel Identification";
      choice type {
        description "Tunnel information type";
        case tunnel-p2p {
          leaf p2p-id {
            type tunnel-ref;
            description "P2P TE tunnel";
          }
        }
        case tunnel-p2mp {
          leaf p2mp-id {
            type tunnel-p2mp-ref;
            description "P2MP TE tunnel";
          }
        }
      }
    }
  }
  output {
    container result {
      description
        "The container result of the RPC operation";
      leaf result {
        type enumeration {
          enum success {
            description "Origin ingress";
          }
          enum in-progress {
            description "Origin egress";
          }
          enum fail {
            description "transit";
          }
        }
      }
    }
  }
}
The TE device YANG module "ietf-te-device" imports the following module(s):

- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-interfaces defined in [RFC8343]
- ietf-routing-types defined in [RFC8294]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]
- ietf-te defined in this document

<CODE BEGINS> file "ietf-te-device@2019-04-09.yang"
module ietf-te-device {
  yang-version 1.1;

  /* Replace with IANA when assigned */
  prefix "te-dev";

  /* Import TE generic types */
  import ietf-te {
    prefix te;
    reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic

/* Import TE generic types */
import ietf-te-types {
    prefix te-types;
    reference "draft-ietf-teas-yang-te-types: A YANG Data Model for
            Common Traffic Engineering Types";
}

import ietf-interfaces {
    prefix if;
    reference "RFC8343: A YANG Data Model for Interface Management";
}

import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
}

import ietf-routing-types {
    prefix "rt-types";
    reference "RFC8294: Common YANG Data Types for the Routing Area";
}

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

contact
    "WG Web: <http://tools.ietf.org/wg/teas/>
    WG List: <mailto:teas@ietf.org>
    WG Chair: Lou Berger
        <mailto:lberger@labn.net>
    WG Chair: Vishnu Pavan Beeram
        <mailto:vbeeram@juniper.net>
    Editor: Tarek Saad
        <mailto:tsaad@juniper.net>
    Editor: Rakesh Gandhi
        <mailto:rgandhi@cisco.com>
    Editor: Vishnu Pavan Beeram
        <mailto:vbeeram@juniper.net>
/**
 * TE LSP device state grouping
 */
grouping lsps-device-state {
  description "TE LSP device state grouping";
  container lsp-timers {
    when "../te:origin-type = 'ingress'" {
      description "Applicable to ingress LSPs only"
    }
    description "Ingress LSP timers";
    leaf life-time {
      type uint32;
      units seconds;
      description "lsp life time"
    }
    leaf time-to-install {
      type uint32;
      units seconds;
      description "lsp installation delay time"
    }
    leaf time-to-destroy {
      type uint32;
      units seconds;
      description
    }
  }
}

description
"YANG data module for TE device configurations, state, RPC and notifications.";

revision "2019-04-09" {
  description "Latest update to TE device YANG module.";
  reference
  "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels and Interfaces"
}

Editor:   Himanshu Shah
<mailto:hshah@ciena.com>

Editor:   Xufeng Liu
<mailto: xufeng.liu.ietf@gmail.com>

Editor:   Igor Bryskin
<mailto:Igor.Bryskin@huawei.com>";
"lsp expiration delay time";
}
}

container downstream-info {
    when "./te:origin-type != 'egress'" {
        description "Applicable to ingress LSPs only";
    }
    description "downstream information";

    leaf nhop {
        type inet:ip-address;
        description "downstream nexthop.";
    }

    leaf outgoing-interface {
        type if:interface-ref;
        description "downstream interface.";
    }

    leaf neighbor {
        type inet:ip-address;
        description "downstream neighbor.";
    }

    leaf label {
        type rt-types:generalized-label;
        description "downstream label.";
    }
}

container upstream-info {
    when "./te:origin-type != 'ingress'" {
        description "Applicable to non-ingress LSPs only";
    }
    description "upstream information";

    leaf phop {
        type inet:ip-address;
        description "upstream nexthop or previous-hop.";
    }
}
leaf neighbor {
    type inet:ip-address;
    description "upstream neighbor.";
}

leaf label {
    type rt-types:generalized-label;
    description "upstream label.";
}

/**
 * Device general groupings.
 */
grouping tunnel-device-config {
    description "Device TE tunnel configs";
    leaf path-invalidation-action {
        type identityref {
            base te-types:path-invalidation-action-type;
        }
        description "Tunnel path invalidation action";
    }
}

grouping lsp-device-timers-config {
    description "Device TE LSP timers configs";
    leaf lsp-install-interval {
        type uint32;
        units seconds;
        description "LSP installation delay time";
    }
    leaf lsp-cleanup-interval {
        type uint32;
        units seconds;
        description "LSP cleanup delay time";
    }
    leaf lsp-invalidation-interval {
        type uint32;
        units seconds;
        description "LSP path invalidation before taking action delay time";
    }
}
/**
 * TE global device generic groupings
 */

/* TE interface container data */
grouping interfaces-grouping {
    description
    "Interface TE configuration data grouping";
    container interfaces {
        description
        "Configuration data model for TE interfaces.";
        uses te-all-attributes;
        list interface {
            key "interface";
            description "TE interfaces.";
            leaf interface {
                type if:interface-ref;
                description "TE interface name.";
            }
        }
    }
}

/**
 * TE interface device generic groupings
 */
grouping te-admin-groups-config {
    description
    "TE interface affinities grouping";
    choice admin-group-type {
        description
        "TE interface administrative groups representation type";
        case value-admin-groups {
            choice value-admin-group-type {
                description "choice of admin-groups";
                case admin-groups {
                    description
                    "Administrative group/Resource class/Color.";
                    leaf admin-group {
                        type te-types:admin-group;
                        description "TE interface administrative group";
                    }
                }
            }
        }
    }
}
case extended-admin-groups {
  if-feature te-types:extended-admin-groups;
  description
    "Extended administrative group/Resource
     class/Color.";
  leaf extended-admin-group {
    type te-types:extended-admin-group;
    description
      "TE interface extended administrative group";
  }
}

}/ TE interface SRLGs */
grouping te-srlgs-config {
  description "TE interface SRLG grouping";
  choice srlg-type {
    description "Choice of SRLG configuration";
    case value-srLgs {
      list values {
        key "value";
        description "List of SRLG values that
                     this link is part of.";
        leaf value {
          type uint32 {
            range "0..4294967295";
          }
grouping te-igp-flooding-bandwidth-config {
  description
  "Configurable items for igp flooding bandwidth threshold configuration.";
  leaf threshold-type {
    type enumeration {
      enum DELTA {
        description
        "DELTA indicates that the local system should flood IGP updates when a change in reserved bandwidth \(\geq\) the specified delta occurs on the interface.";
      }
      enum THRESHOLD_CROSSED {
        description
        "THRESHOLD-CROSSED indicates that the local system should trigger an update (and hence flood) the reserved bandwidth when the reserved bandwidth changes such that it crosses, or becomes equal to one of the threshold values.";
      }
    }
    description
  }
}
"The type of threshold that should be used to specify the values at which bandwidth is flooded. DELTA indicates that the local system should flood IGP updates when a change in reserved bandwidth \( \geq \) the specified delta occurs on the interface. Where THRESHOLD_CROSSED is specified, the local system should trigger an update (and hence flood) the reserved bandwidth when the reserved bandwidth changes such that it crosses, or becomes equal to one of the threshold values;"

leaf delta-percentage {
  when "../threshold-type = 'DELTA'" {
    description
    "The percentage delta can only be specified when the threshold type is specified to be a percentage delta of the reserved bandwidth";
  }
  type rt-types:percentage;
  description
  "The percentage of the maximum-reservable-bandwidth considered as the delta that results in an IGP update being flooded";
}

leaf threshold-specification {
  when "../threshold-type = 'THRESHOLD_CROSSED'" {
    description
    "The selection of whether mirrored or separate threshold values are to be used requires user specified thresholds to be set";
  }
  type enumeration {
    enum MIRRORED_UP_DOWN {
      description
      "MIRRORED_UP_DOWN indicates that a single set of threshold values should be used for both increasing and decreasing bandwidth when determining whether to trigger updated bandwidth values to be flooded in the IGP TE extensions."
    }
    enum SEPARATE_UP_DOWN {
      description
      "SEPARATE_UP_DOWN indicates that a separate threshold values should be used for the increasing and decreasing bandwidth when determining whether to trigger updated bandwidth values to be flooded in the IGP TE extensions.";
    }
  }
}
description
"This value specifies whether a single set of threshold values should be used for both increasing and decreasing bandwidth when determining whether to trigger updated bandwidth values to be flooded in the IGP TE extensions. MIRRORED-UP-DOWN indicates that a single value (or set of values) should be used for both increasing and decreasing values, where SEPARATE-UP-DOWN specifies that the increasing and decreasing values will be separately specified";

leaf-list up-thresholds {
  when "./threshold-type = 'THRESHOLD_CROSSED'" +
    "and "./threshold-specification = 'SEPARATE_UP_DOWN'" {
    description
    "A list of up-thresholds can only be specified when the bandwidth update is triggered based on crossing a threshold and separate up and down thresholds are required";
  }
  type rt-types:percentage;
  description
  "The thresholds (expressed as a percentage of the maximum reservable bandwidth) at which bandwidth updates are to be triggered when the bandwidth is increasing.";
}

leaf-list down-thresholds {
  when "./threshold-type = 'THRESHOLD_CROSSED'" +
    "and "./threshold-specification = 'SEPARATE_UP_DOWN'" {
    description
    "A list of down-thresholds can only be specified when the bandwidth update is triggered based on crossing a threshold and separate up and down thresholds are required";
  }
  type rt-types:percentage;
  description
  "The thresholds (expressed as a percentage of the maximum reservable bandwidth) at which bandwidth updates are to be triggered when the bandwidth is decreasing.";
}

leaf-list up-down-thresholds {
  when "./threshold-type = 'THRESHOLD_CROSSED'" +
    "and "./threshold-specification = 'MIRRORED_UP_DOWN'" {
    description

"A list of thresholds corresponding to both increasing and decreasing bandwidths can be specified only when an update is triggered based on crossing a threshold, and the same up and down thresholds are required.";

}  
  type rt-types:percentage;  
  description  
  "The thresholds (expressed as a percentage of the maximum reservable bandwidth of the interface) at which bandwidth updates are flooded - used both when the bandwidth is increasing and decreasing";
  }
}

/* TE interface metric */
grouping te-metric-config {  
  description "Interface TE metric grouping";
  leaf te-metric {  
    type te-types:te-metric;  
    description "Interface TE metric.";
  }
}

/* TE interface switching capabilities */
grouping te-switching-cap-config {  
  description "TE interface switching capabilities";
  list switching-capabilities {  
    key "switching-capability";
    description "List of interface capabilities for this interface";
    leaf switching-capability {  
      type identityref {  
        base te-types:switching-capabilities;  
      }  
      description "Switching Capability for this interface";
    }
    leaf encoding {  
      type identityref {  
        base te-types:lsp-encoding-types;  
      }  
      description "Encoding supported by this interface";
    }
  }
}
grouping te-advertisements-state {
    description "TE interface advertisements state grouping";
    container te-advertisements-state {
        description "TE interface advertisements state container";
        leaf flood-interval {
            type uint32;
            description "The periodic flooding interval";
        }
        leaf last-flooded-time {
            type uint32;
            units seconds;
            description "Time elapsed since last flooding in seconds";
        }
        leaf next-flooded-time {
            type uint32;
            units seconds;
            description "Time remained for next flooding in seconds";
        }
        leaf last-flooded-trigger {
            type enumeration {
                enum link-up {
                    description "Link-up flooding trigger";
                }
                enum link-down {
                    description "Link-up flooding trigger";
                }
                enum threshold-up {
                    description "Bandwidth reservation up threshold";
                }
                enum threshold-down {
                    description "Bandwidth reservation down threshold";
                }
                enum bandwidth-change {
                    description "Bandwidth capacity change";
                }
                enum user-initiated {
                    description "Initiated by user";
                }
                enum srlg-change {
                    description "SRLG property change";
                }
            }
        }
    }
}
enum periodic-timer {
    description "Periodic timer expired";
}

default 'periodic-timer';
description "Trigger for the last flood";

list advertized-level-areas {
    key level-area;
    description "List of areas the TE interface is advertised in";
    leaf level-area {
        type uint32;
        description "The IGP area or level where the TE interface state is advertised in";
    }
}

/* TE interface attributes grouping */
grouping te-attributes {
    description "TE attributes configuration grouping";
    uses te-metric-config;
    uses te-admin-groups-config;
    uses te-srlgs-config;
    uses te-igp-flooding-bandwidth-config;
    uses te-switching-cap-config;
    container state {
        config false;
        description "State parameters for interface TE metric";
        uses te-advertisements-state;
    }
}

grouping te-all-attributes {
    description "TE attributes configuration grouping for all interfaces";
    uses te-igp-flooding-bandwidth-config;
}

/*** End of TE interfaces device groupings ***/
* TE device augmentations

/*
augment "/te:te" {
    description "TE global container.";
    /* TE Interface Configuration Data */
    uses interfaces-grouping;
    container performance-thresholds {
        description
        "Performance parameters configurable thresholds";
    }
}
*/

/* TE globals device augmentation */
augment "/te:te/te:globals" {
    description
    "Global TE device specific configuration parameters";
    uses lsp-device-timers-config;
}

/* TE tunnels device configuration augmentation */
augment "/te:te/te:tunnels/te:tunnel" {
    description
    "Tunnel device dependent augmentation";
    uses lsp-device-timers-config;
}

/* TE LSPs device state augmentation */
augment "/te:te/te:lsps-state/te:lsp" {
    description
    "LSP device dependent augmentation";
    uses lsps-device-state;
}

    description
    "LSP device dependent augmentation";
    uses lsps-device-state;
}

    description
    "LSP device dependent augmentation";
    uses lsps-device-state;
}

/* TE interfaces RPCs/execution Data */
rpc interfaces-rpc {
    description
    "Execution data for TE interfaces."
}

/* TE Interfaces Notification Data */
notification interfaces-notif {
    description
    "Notification messages for TE interfaces."
}

<CODE ENDS>

Figure 8: TE device specific YANG module

5. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested to be made.

XML: N/A, the requested URI is an XML namespace.

XML: N/A, the requested URI is an XML namespace.

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

name:       ietf-te
prefix:     ietf-te
reference:  RFCXXXX

name:       ietf-te-device
prefix:     ietf-te-device
reference:  RFCXXXX

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF
users to a pre-configured subset of all available NETCONF protocol operations and content.

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., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations. Following are the subtrees and data nodes and their sensitivity/vulnerability:

"/te/globals": This module specifies the global TE configurations on a device. Unauthorized access to this container could cause the device to ignore packets it should receive and process.

"/te/tunnels": This list specifies the configured TE tunnels on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

"/te/lsps-state": This list specifies the state derived LSPs. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

"/te/interfaces": This list specifies the configured TE interfaces on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

7. Acknowledgement

The authors would like to thank the members of the multi-vendor YANG design team who are involved in the definition of this model.

The authors would also like to thank Loa Andersson, Lou Berger, Sergio Belotti, Italo Busi, Carlo Perocchio, Francesco Lazzeri, Aihua Guo, Dhruv Dhody, Anurag Sharma, and Xian Zhang for their comments and providing valuable feedback on this document.

8. Contributors
9. References

9.1. Normative References

[I-D.ietf-teas-yang-path-computation]

[I-D.ietf-teas-yang-rsvp]

[I-D.ietf-teas-yang-te-types]


9.2. Informative References


Authors’ Addresses

Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems Inc
Email: rgandhi@cisco.com

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

A YANG Data Model for MPLS Traffic Engineering Tunnels
draft-ietf-teas-yang-te-mpls-01

Abstract

This document defines a YANG data model for the configuration and management of Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) tunnels, Label Switched Paths (LSPs) and interfaces. The model augments the TE generic YANG model for MPLS packet dataplane technology.

This model covers data for configuration, operational state, remote procedural calls, and event notifications.

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 August 27, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

YANG [RFC6020] and [RFC7950] is a data modeling language used to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG has proved relevant beyond its initial confines, as bindings to other interfaces (e.g. RESTCONF [RFC8040]) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document describes the YANG data model for configuration and management of MPLS TE tunnels, LSPs, and interfaces. Other YANG module(s) that model the establishment of MPLS LSP(s) via signaling protocols such as RSVP-TE ([RFC3209], [RFC3473]) are described in separate document(s).

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP
1.2. Prefixes in Data Node Names

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>rt-types</td>
<td>ietf-routing-types</td>
<td>[RFC8294]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>te-dev</td>
<td>ietf-te-device</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>te-mpls</td>
<td>ietf-te-mpls</td>
<td>This document</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[I-D.ietf-teas-yang-te-types]</td>
</tr>
<tr>
<td>te-mpls-types</td>
<td>ietf-te-mpls-types</td>
<td>[I-D.ietf-teas-yang-te-types]</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

1.3. Acronyms and Abbreviations

MPLS: Multiprotocol Label Switching
LSP: Label Switched Path
LSR: Label Switching Router
LER: Label Edge Router
TE: Traffic Engineering

2. MPLS TE YANG Model

The MPLS TE YANG model covers the configuration, state, RPC and notifications data pertaining to MPLS TE interfaces, tunnels and LSPs parameters. The data specific to the signaling protocol used to establish MPLS LSP(s) is outside the scope of this document and is covered in other documents, e.g. in [I-D.ietf-teas-yang-rsvp] and [I-D.ietf-teas-yang-rsvp-te].

2.1. Module(s) Relationship

The MPLS TE YANG module "ietf-te-mpls" imports the following modules:

- ietf-te and ietf-te-device defined in [I-D.ietf-teas-yang-te]
o ietf-te-types and ietf-te-packet-types defined in [I-D.ietf-teas-yang-te-types]

o ietf-routing-types defined in [RFC8294]

o ietf-mpls-static defined in [I-D.ietf-mpls-static-yang]

TE generic
    +---------+
    |         o: augment
    | ietf-te |
    +---------+
              o o
              o
              +-----+  +-----+
              |              |
              +--------------+   +--------------+
              | RSVP-TE | ietf-rsvp-te |   | ietf-te-mpls |
              +--------------+   +--------------+

Figure 1: Relationship of MPLS TE module with TE generic and RSVP-TE YANG modules

The MPLS TE YANG module "ietf-te-mpls" augments the "ietf-te" TE generic YANG module as shown in Figure 1.

2.2. Model Tree Diagram

Figure 2 shows the tree diagram of the MPLS TE YANG model that is defined in ietf-te-mpls.yang.

module: ietf-te-mpls
augment /te:te/te-dev:performance-thresholds:
  |--rw throttle
    |--rw one-way-delay-offset?         uint32
    |--rw measure-interval?             uint32
    |--rw advertisement-interval?       uint32
    |--rw suppression-interval?         uint32
    |--rw threshold-out
      |--rw one-way-delay?               uint32
      |--rw one-way-residual-bandwidth?  rt-types:bandwidth-ieee-float32
      |--rw one-way-available-bandwidth? rt-types:bandwidth-ieee-float32
      |--rw one-way-utilized-bandwidth?  rt-types:bandwidth-ieee-float32
      |--rw two-way-delay?               uint32
      |--rw one-way-min-delay?           uint32
      |--rw one-way-max-delay?           uint32
augment /te:te/te:tunnels/te:tunnel:
  +--rw tunnel-igp-shortcut
     |  +--rw shortcut-eligible?   boolean
     |  +--rw metric-type?         identityref
     |  +--rw metric?              int32
     |  +--rw routing-afs*         inet:ip-version
  +--rw forwarding
     |  +--rw binding-label?       rt-types:mpls-label

---rw one-way-delay-variation?       uint32
---rw one-way-packet-loss?           decimal64
---rw two-way-min-delay?             uint32
---rw two-way-max-delay?             uint32
---rw two-way-delay-variation?       uint32
---rw two-way-packet-loss?           decimal64
---rw threshold-in
  +--rw one-way-delay?       uint32
  |   +--rw one-way-residual-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  |   +--rw one-way-available-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  |   +--rw one-way-utilized-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  +--rw one-way-min-delay?             uint32
  +--rw one-way-max-delay?             uint32
  +--rw one-way-delay-variation?       uint32
  +--rw one-way-packet-loss?           decimal64
  +--rw two-way-min-delay?             uint32
  +--rw two-way-max-delay?             uint32
  +--rw two-way-delay-variation?       uint32
  +--rw two-way-packet-loss?           decimal64
---rw threshold-accelerated-advertisement
  +--rw one-way-delay?       uint32
  |   +--rw one-way-residual-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  |   +--rw one-way-available-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  |   +--rw one-way-utilized-bandwidth?
  |       rt-types:bandwidth-ieee-float32
  +--rw one-way-min-delay?             uint32
  +--rw one-way-max-delay?             uint32
  +--rw one-way-delay-variation?       uint32
  +--rw one-way-packet-loss?           decimal64
  +--rw two-way-min-delay?             uint32
  +--rw two-way-max-delay?             uint32
  +--rw two-way-delay-variation?       uint32
  +--rw two-way-packet-loss?           decimal64
---rw load-share?       uint32
---rw policy-class?     uint8
++-rw bandwidth-mpls
    +--rw specification-type?
        |  te-packet-types:te-bandwidth-requested-type
    +--rw set-bandwidth?      te-packet-types:bandwidth-kbps
    +--rw class-type?         te-types:te-ds-class
    +--ro state
        |  +--ro signaled-bandwidth?   te-packet-types:bandwidth-kbps
    +--rw auto-bandwidth
        +--rw enabled?            boolean
        +--rw min-bw?             te-packet-types:bandwidth-kbps
        +--rw max-bw?             te-packet-types:bandwidth-kbps
        +--rw adjust-interval?    uint32
        +--rw adjust-threshold?   rt-types:percentage
        +--rw overflow
            |  +--rw enabled?               boolean
            |  +--rw overflow-threshold?   rt-types:percentage
            |  +--rw trigger-event-count?    uint16
        +--rw underflow
            |  +--rw enabled?               boolean
            |  +--rw underflow-threshold?   rt-types:percentage
            |  +--rw trigger-event-count?    uint16
augment /te:te/te:tunnels/te:tunnel/te:p2p-primary-paths
    /te:p2p-primary-path:
    +--rw static-lsp-name?   mpls-static:static-lsp-ref
augment /te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths
    /te:p2p-secondary-path:
    +--rw static-lsp-name?   mpls-static:static-lsp-ref
augment /te:te/te:globals/te:named-path-constraints
    /te:named-path-constraint:
    +--rw bandwidth
        +--rw specification-type?
            |  te-packet-types:te-bandwidth-requested-type
        +--rw set-bandwidth?      te-packet-types:bandwidth-kbps
        +--rw class-type?         te-types:te-ds-class
        +--ro state
            |  +--ro signaled-bandwidth?   te-packet-types:bandwidth-kbps
augment /te:te/te:tunnels/te:tunnel/te:p2p-primary-paths
    /te:p2p-primary-path/te:lsps/te:lsp:
    +--ro performance-metrics-one-way
        |  +--ro one-way-delay?       uint32
        |  +--ro one-way-delay-normality?
            |    te-types:performance-metrics-normality
        |  +--ro one-way-residual-bandwidth?
            |    rt-types:bandwidth-ieee-float32
        |  +--ro one-way-residual-bandwidth-normality?
            |    te-types:performance-metrics-normality
Saad, et al. Expires August 27, 2019
---ro one-way-available-bandwidth?
   rt-types:bandwidth-ieee-float32
---ro one-way-available-bandwidth-normality?
   te-types:performance-metrics-normality
---ro one-way-utilized-bandwidth?
   rt-types:bandwidth-ieee-float32
---ro one-way-utilized-bandwidth-normality?
   te-types:performance-metrics-normality
---ro one-way-min-delay?                       uint32
---ro one-way-min-delay-normality?
   te-types:performance-metrics-normality
---ro one-way-max-delay?                       uint32
---ro one-way-max-delay-normality?
   te-types:performance-metrics-normality
---ro one-way-delay-variation?                 uint32
---ro one-way-delay-variation-normality?
   te-types:performance-metrics-normality
---ro one-way-packet-loss?                     decimal64
---ro one-way-packet-loss-normality?
   te-types:performance-metrics-normality
---ro performance-metrics-two-way
---ro two-way-delay?                       uint32
---ro two-way-delay-normality?
   te-types:performance-metrics-normality
---ro two-way-min-delay?                       uint32
---ro two-way-min-delay-normality?
   te-types:performance-metrics-normality
---ro two-way-max-delay?                       uint32
---ro two-way-max-delay-normality?
   te-types:performance-metrics-normality
---ro two-way-delay-variation?                 uint32
---ro two-way-delay-variation-normality?
   te-types:performance-metrics-normality
---ro two-way-packet-loss?                     decimal64
---ro two-way-packet-loss-normality?
   te-types:performance-metrics-normality

Figure 2: MPLS TE model configuration and state tree

2.3. MPLS TE YANG Module

<CODE BEGINS> file "ietf-te-mpls@2019-02-23.yang"
module ietf-te-mpls {
   yang-version 1.1;

   /* Replace with IANA when assigned */
   prefix "te-mpls";

   Saad, et al.             Expires August 27, 2019                

/* Import TE base model */
import ietf-te {
    prefix te;
    reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

import ietf-te-device {
    prefix te-dev;
    reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
}

/* Import TE MPLS types */
import ietf-te-packet-types {
    prefix "te-packet-types";
    reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering Types";
}

/* Import TE generic types */
import ietf-te-types {
    prefix te-types;
    reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering Types";
}

/* Import routing types */
import ietf-routing-types {
    prefix rt-types;
    reference "RFC8294: Common YANG Data Types for the Routing Area";
}

import ietf-mpls-static {
    prefix mpls-static;
    reference "draft-ietf-mpls-static-yang: A YANG Data Model for MPLS Static LSPs";
}

import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
}

organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS) Working Group";
YANG data module for MPLS TE configurations, state, RPC and notifications. The model fully conforms to the Network Management Datastore Architecture (NMDA).

Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (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.

// RFC Ed.: replace XXXX with actual RFC number and remove this note.

// RFC Ed.: update the date below with the date of RFC publication // and remove this note.
revision "2019-02-23" {
    description "Latest update to MPLS TE YANG module.";
    reference
        "RFCXXXX: A YANG Data Model for MPLS-TE Tunnels and LSP(s)";
}

/* MPLS TE tunnel properties*/
grouping tunnel-igp-shortcut-config {
    description "TE tunnel IGP shortcut configs";
    leaf shortcut-eligible {
        type boolean;
        default "true";
        description
            "Whether this LSP is considered to be eligible for us as a
            shortcut in the IGP. In the case that this leaf is set to
            true, the IGP SPF calculation uses the metric specified to
            determine whether traffic should be carried over this LSP";
    }
    leaf metric-type {
        type identityref {
            base te-types:lsp-metric-type;
        }
        default te-types:lsp-metric-inherited;
        description
            "The type of metric specification that should be used to set
            the LSP(s) metric";
    }
    leaf metric {
        type int32;
        description
            "The value of the metric that should be specified. The value
            supplied in this leaf is used in conjunction with the metric
            type to determine the value of the metric used by the system.
            Where the metric-type is set to lsp-metric-absolute - the
            value of this leaf is used directly; where it is set to
            lsp-metric-relative, the relevant (positive or negative)
            offset is used to formulate the metric; where metric-type
            is lsp-metric-inherited, the value of this leaf is not
            utilized";
    }
    leaf-list routing-afs {
        type inet:ip-version;
        description
            "Address families";
    }
}
grouping tunnel-igp-shortcuts {
  description "TE tunnel IGP shortcut grouping";
  container tunnel-igp-shortcut {
    description "Tunnel IGP shortcut properties";
    uses tunnel-igp-shortcut-config;
  }
}

grouping tunnel-forwarding-adjacency-configs {
  description "Tunnel forwarding adjacency grouping";
  leaf binding-label {
    type rt-types:mpls-label;
    description "MPLS tunnel binding label";
  }
  leaf load-share {
    type uint32 {
      range "1..4294967295";
    }
    description "ECMP tunnel forwarding load-share factor.";
  }
  leaf policy-class {
    type uint8 {
      range "1..7";
    }
    description "The class associated with this tunnel";
  }
}

grouping tunnel-forwarding-adjacency {
  description "Properties for using tunnel in forwarding.";
  container forwarding {
    description "Tunnel forwarding properties container";
    uses tunnel-forwarding-adjacency-configs;
  }
}

/*** End of MPLS TE tunnel configuration/state */
grouping te-lsp-auto-bandwidth-config {
  description "Configuration parameters related to autobandwidth";
  leaf enabled {
    type boolean;
  }
}
leaf min-bw {
  type te-packet-types:bandwidth-kbps;
  description
    "set the minimum bandwidth in Kbps for an auto-bandwidth LSP";
}

leaf max-bw {
  type te-packet-types:bandwidth-kbps;
  description
    "set the maximum bandwidth in Kbps for an auto-bandwidth LSP";
}

leaf adjust-interval {
  type uint32;
  description
    "time in seconds between adjustments to LSP bandwidth";
}

leaf adjust-threshold {
  type rt-types:percentage;
  description
    "percentage difference between the LSP’s specified bandwidth and its current bandwidth allocation -- if the difference is greater than the specified percentage, auto-bandwidth adjustment is triggered";
}

grouping te-lsp-overflow-config {
  description
    "configuration for MPLS LSP bandwidth overflow adjustment";

  leaf enabled {
    type boolean;
    default false;
    description
      "Enables MPLS LSP bandwidth overflow";
  }
}
leaf overflow-threshold {
  type rt-types:percentage;
  description
  "bandwidth percentage change to trigger
   an overflow event";
}

leaf trigger-event-count {
  type uint16;
  description
  "number of consecutive overflow sample
   events needed to trigger an overflow adjustment";
}
}
grouping te-lsp-underflow-config {
  description
  "configuration for MPLS LSP bandwidth
   underflow adjustment";

  leaf enabled {
    type boolean;
    default false;
    description
    "enables bandwidth underflow
     adjustment on the LSP";
  }

  leaf underflow-threshold {
    type rt-types:percentage;
    description
    "bandwidth percentage change to trigger
     and underflow event";
  }

  leaf trigger-event-count {
    type uint16;
    description
    "number of consecutive underflow sample
     events needed to trigger an underflow adjustment";
  }
}
grouping te-tunnel-bandwidth-config {
  description

  leaf overflow-threshold {
    type rt-types:percentage;
    description
    "bandwidth percentage change to trigger
     an overflow event";
  }

  leaf trigger-event-count {
    type uint16;
    description
    "number of consecutive overflow sample
     events needed to trigger an overflow adjustment";
  }

  leaf underflow-threshold {
    type rt-types:percentage;
    description
    "bandwidth percentage change to trigger
     and underflow event";
  }

  leaf trigger-event-count {
    type uint16;
    description
    "number of consecutive underflow sample
     events needed to trigger an underflow adjustment";
  }
}
"Configuration parameters related to bandwidth for a tunnel";

leaf specification-type {
  type te-packet-types:te-bandwidth-requested-type;
  default specified;
  description
    "The method used for setting the bandwidth, either explicitly
       specified or configured";
}

leaf set-bandwidth {
  when "../specification-type = 'specified'" {
    description
      "The bandwidth value when bandwidth is explicitly
         specified";
  }
  type te-packet-types:bandwidth-kbps;
  description
    "set bandwidth explicitly, e.g., using
     offline calculation";
}

leaf class-type {
  type te-types:te-ds-class;
  description
    "The Class-Type of traffic transported by the LSP.";
  reference "RFC4124: section-4.3.1";
}

grouping te-tunnel-bandwidth-state {
  description
    "Operational state parameters relating to bandwidth for a tunnel";
  leaf signaled-bandwidth {
    type te-packet-types:bandwidth-kbps;
    description
      "The currently signaled bandwidth of the LSP. In the case where
       the bandwidth is specified explicitly, then this will match the
       value of the set-bandwidth leaf; in cases where the bandwidth is
       dynamically computed by the system, the current value of the
       bandwidth should be reflected.";
  }
}

grouping tunnel-bandwidth_top {
  description
    "Top level grouping for specifying bandwidth for a tunnel";
}
container bandwidth-mpls {
  description  
    "Bandwidth configuration for TE LSPs";

  uses te-tunnel-bandwidth-config;

  container state {
    config false;
    description  
      "State parameters related to bandwidth 
      configuration of TE tunnels";
    uses te-tunnel-bandwidth-state;
  }

  container auto-bandwidth {
    when ".../specification-type = 'auto'" {
      description  
        "Include this container for auto bandwidth 
        specific configuration";
    }
    description  
      "Parameters related to auto-bandwidth";

    uses te-lsp-auto-bandwidth-config;

    container overflow {
      description  
        "configuration of MPLS overflow bandwidth 
        adjustment for the LSP";

      uses te-lsp-overflow-config;
    }

    container underflow {
      description  
        "configuration of MPLS underflow bandwidth 
        adjustment for the LSP";

      uses te-lsp-underflow-config;
    }
  }
}

grouping te-path-bandwidth_top {
  description  
    "Top level grouping for specifying bandwidth for a TE path";
container bandwidth {
  description
  "Bandwidth configuration for TE LSPs";

  uses te-tunnel-bandwidth-config;
  container state {
    config false;
    description
    "State parameters related to bandwidth
    configuration of TE tunnels";
    uses te-tunnel-bandwidth-state;
  }
}

/**
 * MPLS TE augmentations
 */

augment "/te:te/te-dev:performance-thresholds" {
  uses te-packet-types:performance-metrics-throttle-container-packet;
  description
  "Performance parameters configurable thresholds";
}

/* MPLS TE interface augmentations */

/* MPLS TE tunnel augmentations */

augment "/te:te/te:tunnels/te:tunnel" {
  description "MPLS TE tunnel config augmentations";
  uses tunnel-igp-shortcuts;
  uses tunnel-forwarding-adjacency;
  uses tunnel-bandwidth_top;
}

/* MPLS TE LSPs augmentations */

augment "/te:te/te:tunnels/te:tunnel/" +
  "te:p2p-primary-paths/te:p2p-primary-path" {
  when "/te:te/te:tunnels/te:tunnel" +
  "/te:p2p-primary-paths/te:p2p-primary-path" +
  "/te:path-setup-protocol = 'te-types:path-setup-static'" {
    description
    "When the path is statically provisioned";
  }
  description "MPLS TE LSP augmentation";
  leaf static-lsp-name {
type mpls-static:static-lsp-ref;
description "Static LSP name";
}

augment "/te:te/te:tunnels/te:tunnel/" +
"te:p2p-secondary-paths/te:p2p-secondary-path" {
when "/te:te/te:tunnels/te:tunnel" +
"/te:p2p-secondary-paths/te:p2p-secondary-path/" +
"te:path-setup-protocol = 'te-types:path-setup-static'" {
  description
  "When the path is statically provisioned";
}
description "MPLS TE LSP augmentation";
leaf static-lsp-name {
  type mpls-static:static-lsp-ref;
  description "Static LSP name";
}
}

augment "/te:te/te:globals/te:named-path-constraints/" +
"te:named-path-constraint" {
  description "foo";
  uses te-path-bandwidth_top;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" +
"/te:p2p-primary-path/te:lsps/te:lsp" {
  description
  "MPLS TE generic data augmentation pertaining to specific TE LSP";
  uses te-packet-types:performance-metrics-attributes-packet;
}
<CODE ENDS>

Figure 3: TE generic YANG module

3. IANA Considerations

This document registers the following URIs in the IETF XML registry
[RFC3688]. Following the format in [RFC3688], the following
registration is requested to be made.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name:       ietf-te-mpls
prefix:     ietf-te-mpls
reference:  RFC3209

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

A number of data nodes defined in this YANG module 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., <edit-config>) to these data nodes without proper protection can have a negative effect on MPLS network operations. Following are the subtrees and data nodes and their sensitivity/vulnerability:

"/te/tunnels": The augmentation to this list specifies configuration to TE tunnels on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

"/te/globals": The augmentation to this target specifies configuration applicable to the to all or one TE device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

5. Contributors

Himanshu Shah
Ciena
Email: hshah@ciena.com

6. Normative References
[I-D.ietf-mpls-static-yang]

[I-D.ietf-teas-yang-rsvp]

[I-D.ietf-teas-yang-rsvp-te]

[I-D.ietf-teas-yang-te]

[I-D.ietf-teas-yang-te-types]


Authors’ Addresses

Tarek Saad
Cisco Systems Inc

Email: tsaad@cisco.com

May 8, 2019

YANG models for VN & TE Performance Monitoring Telemetry and Scaling Intent Autonomics
draft-lee-teas-actn-pm-telemetry-autonomics-17

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on November 8, 2019.

Copyright Notice

Abstract

This document provides YANG data models that describe performance monitoring telemetry and scaling intent mechanism for TE-tunnels and Virtual Networks (VN).

The models presented in this draft allow customers to subscribe to and monitor their key performance data of their interest on the level of TE-tunnel or VN. The models also provide customers with the ability to program autonomic scaling intent mechanism on the level of TE-tunnel as well as VN.
1. Introduction

The YANG model discussed in [VN] is used to operate customer-driven Virtual Networks (VNs) during the VN instantiation, VN computation, and its life-cycle service management and operations. YANG model discussed in [TE-Tunnel] is used to operate TE-tunnels during the tunnel instantiation, and its life-cycle management and operations.

The models presented in this draft allow the applications hosted by the customers to subscribe to and monitor their key performance data of their interest on the level of VN [VN] or TE-tunnel [TE-Tunnel]. The key characteristic of the models presented in this document is a top-down programmability that allows the applications hosted by the customers to subscribe to and monitor key performance data of their interest and autonomic scaling intent mechanism on the level of VN as well as TE-tunnel.

According to the classification of [RFC8309], the YANG data models presented in this document can be classified as customer service models, which is mapped to CMI (Customer Network Controller (CNC)-Multi-Domain Service Coordinator (MSDC) interface) of ACTN [RFC8453].

[RFC8233] describes key network performance data to be considered for end-to-end path computation in TE networks. Key performance indicator (KPI) is a term that describes critical performance data that may affect VN/TE-tunnel service. The services provided can be optimized to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers.

This document provides YANG data models generically applicable to any VN/TE-Tunnel service clients to provide an ability to program their customized performance monitoring subscription and publication data models and automatic scaling in/out intent data models. These models can be utilized by a client network controller to initiate this capability to a transport network controller communicating with the client controller via a NETCONF [RFC8341] or a RESTCONF [RFC8040] interface.
The term performance monitoring being used in this document is different from the term that has been used in transport networks for many years. Performance monitoring in this document refers to subscription and publication of streaming telemetry data. Subscription is initiated by the client (e.g., CNC) while publication is provided by the network (e.g., MDSC/PNC) based on the client’s subscription. As the scope of performance monitoring in this document is telemetry data on the level of client’s VN or TE-tunnel, the entity interfacing the client (e.g., MDSC) has to provide VN or TE-tunnel level information. This would require controller capability to derive VN or TE-tunnel level performance data based on lower-level data collected via PM counters in the Network Elements (NE). How the controller entity derives such customized level data (i.e., VN or TE-tunnel level) is out of the scope of this document.

The data model includes configuration and state data according to the new Network Management Datastore Architecture [RFC8342].

1.1. Terminology

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

Key Performance Data: This refers to a set of data the customer is interested in monitoring for their instantiated VNs or TE-tunnels. Key performance data and key performance indicators are interchangeable in this draft.

Scaling: This refers to the network ability to re-shape its own resources. Scale out refers to improve network performance by increasing the allocated resources, while scale in refers to decrease the allocated resources, typically because the existing resources are unnecessary.

Scaling Intent: To declare scaling conditions, scaling intent is used. Specifically, scaling intent refers to the intent expressed by the client that allows the client to program/configure conditions of their key performance data either for scaling out or scaling in. Various conditions can be set for scaling intent on either VN or TE-tunnel level.

Network Autonomics: This refers to the network automation capability that allows client to initiate scaling intent mechanisms and provides the client with the status of the adjusted network
resources based on the client’s scaling intent in an automated fashion.

1.2. Tree diagram

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

1.3. Prefixes in Data Node Names

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

+---------+------------------------------+-----------------+
| Prefix  | YANG module                  | Reference       |
|---------+------------------------------+-----------------+
| rt      | ietf-routing-types           | [RFC8294]       |
| te      | ietf-te                      | [TE-Tunnel]     |
| te-types| ietf-te-types                | [TE-Types]      |
| te-tel  | ietf-te-kpi-telemetry        | [This I-D]      |
| vn      | ietf-vn                      | [VN]            |
| vn-tel  | ietf-vn-kpi-telemetry        | [This I-D]      |
+---------+------------------------------+-----------------+

Table 1: Prefixes and corresponding YANG modules

2. Use-Cases

[PERF] describes use-cases relevant to this draft. It introduces the dynamic creation, modification and optimization of services based on the performance monitoring. Figure 1 shows a high-level workflows for dynamic service control based on traffic monitoring.
Some of the key points from [PERF] are as follows:

- Network traffic monitoring is important to facilitate automatic discovery of the imbalance of network traffic, and initiate the network optimization, thus helping the network operator or the virtual network service provider to use the network more efficiently and save the Capital Expense (CAPEX) and the Operating Expense (OPEX).

Customer services have various Service Level Agreement (SLA) requirements, such as service availability, latency, latency jitter, packet loss rate, Bit Error Rate (BER), etc. The transport network can satisfy service availability and BER requirements by providing different protection and restoration mechanisms. However, for other performance parameters, there are no such mechanisms. In order to provide high quality services according to customer SLA, one possible solution is to measure the SLA related performance parameters, and dynamically provision and optimize services based on the performance monitoring results.

Performance monitoring in a large scale network could generate a huge amount of performance information. Therefore, the appropriate way to deliver the information in the client and network interfaces should be carefully considered.

3. Design of the Data Models

The YANG models developed in this document describe two models:

(i) TE KPI Telemetry Model which provides the TE-Tunnel level of performance monitoring mechanism and scaling intent mechanism that allows scale in/out programming by the customer. (See Section 3.1 & 7.1 for details).

(ii) VN KPI Telemetry Model which provides the VN level of the aggregated performance monitoring mechanism and scaling intent mechanism that allows scale in/out programming by the customer (See Section 3.2 & 7.2 for details).

3.1. TE KPI Telemetry Model

This module describes performance telemetry for TE-tunnel model. The telemetry data is augmented to tunnel state. This module also allows autonomic traffic engineering scaling intent configuration mechanism on the TE-tunnel level. Various conditions can be set for auto-scaling based on the telemetry data (See Section 5 for details).

The TE KPI Telemetry Model augments the TE-Tunnel Model to enhance TE performance monitoring capability. This monitoring capability
will facilitate proactive re-optimization and reconfiguration of TEs based on the performance monitoring data collected via the TE KPI Telemetry YANG model.

3.2. VN KPI Telemetry Model

This module describes performance telemetry for VN model. The telemetry data is augmented both at the VN Level as well as individual VN member level. This module also allows autonomic traffic engineering scaling intent configuration mechanism on the VN level. Scale in/out criteria might be used for network autonemics in order the controller to react to a certain set of variations in monitored parameters (See Section 4 for illustrations).

Moreover, this module also provides mechanism to define aggregated telemetry parameters as a grouping of underlying VN level telemetry parameters. Grouping operation (such as maximum, mean) could be set at the time of configuration. For example, if maximum grouping operation is used for delay at the VN level, the VN telemetry data is reported as the maximum (delay_vn_member_1, delay_vn_member_2,.. delay_vn_member_N). Thus, this telemetry abstraction mechanism allows the grouping of a certain common set of telemetry values under a grouping operation. This can be done at the VN-member level to suggest how the E2E telemetry be inferred from the per domain tunnel created and monitored by PNCs. One proposed example is the following:
1. Client sets the grouping op, and subscribes to the VN level telemetry for Delay and Utilized-bw-percentage

2. Orchestrator pushes:
   - VN level telemetry for VN Utilized-bw-percentage (Minimum across VN Members)
   - VN Delay (Maximum across VN Members)

The VN Telemetry Model augments the basic VN model to enhance VN monitoring capability. This monitoring capability will facilitate proactive re-optimization and reconfiguration of VNs based on the performance monitoring data collected via the VN Telemetry YANG model.

4. Autonomic Scaling Intent Mechanism

Scaling intent configuration mechanism allows the client to configure automatic scale-in and scale-out mechanisms on both the TE-tunnel and the VN level. Various conditions can be set for auto-scaling based on the PM telemetry data.

There are a number of parameters involved in the mechanism:

- scale-out-intent or scale-in-intent: whether to scale-out or scale-in.
- performance-type: performance metric type (e.g., one-way-delay, one-way-delay-min, one-way-delay-max, two-way-delay, two-way-delay-min, two-way-delay-max, utilized bandwidth, etc.)
. threshold-value: the threshold value for a certain performance-type that triggers scale-in or scale-out.

. scaling-operation-type: in case where scaling condition can be set with one or more performance types, then scaling-operation-type (AND, OR, MIN, MAX, etc.) is applied to these selected performance types and its threshold values.

. Threshold-time: the duration for which the criteria must hold true.

. Cooldown-time: the duration after a scaling action has been triggered, for which there will be no further operation.

The following tree is a part of ietf-te-kpi-telemetry tree whose model is presented in full detail in Sections 6 & 7.

module: ietf-te-kpi-telemetry
augment /te:te/tunnels/te:tunnel:
  +-rw te-scaling-intent
    |  +-rw scale-in-intent
    |     +-rw threshold-time?           uint32
    |     +-rw cooldown-time?             uint32
    |     +-rw scale-in-operation-type?   scaling-criteria-operation
    |     +-rw scaling-condition* [performance-type]
    |        +-rw performance-type       identityref
    |        +-rw threshold-value?        string
    |        +-rw te-telemetry-tunnel-ref?
    |            -> /te:te/tunnels/tunnel/name

    +-rw scale-out-intent
    |  +-rw threshold-time?           uint32
    |  +-rw cooldown-time?             uint32
    |  +-rw scale-out-operation-type?  scaling-criteria-operation
    |  +-rw scaling-condition* [performance-type]
    |     +-rw performance-type       identityref
    |     +-rw threshold-value?        string
    |     +-rw te-telemetry-tunnel-ref?
    |        -> /te:te/tunnels/tunnel/name

Let say the client wants to set the scaling out operation based on two performance-types (e.g., two-way-delay and utilized-bandwidth for a te-tunnel), it can be done as follows:

. Set Threshold-time: x (sec) (duration for which the criteria must hold true)
Set Cooldown-time: \( y \) (sec) (the duration after a scaling action has been triggered, for which there will be no further operation)

Set AND for the scale-out-operation-type

In the scaling condition’s list, the following two components can be set:

List 1: Scaling Condition for Two-way-delay

- performance type: Two-way-delay
- threshold-value: \( z \) milli-seconds

List 2: Scaling Condition for Utilized bandwidth

- performance type: Utilized bandwidth
- threshold-value: \( w \) megabytes

5. Notification

This model does not define specific notifications. To enable notifications, the mechanism defined in [YANG-PUSH] and [Event-Notification] can be used. This mechanism currently allows the user to:

- Subscribe to notifications on a per client basis.
- Specify subtree filters or xpath filters so that only interested contents will be sent.
- Specify either periodic or on-demand notifications.

5.1. YANG Push Subscription Examples

[YANG-PUSH] allows subscriber applications to request a continuous, customized stream of updates from a YANG datastore.

Below example shows the way for a client to subscribe to the telemetry information for a particular tunnel (Tunnell). The telemetry parameter that the client is interested in is one-way-delay.

<netconf:rpc netconf:message-id="101"
  xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push:1.0">
    <filter netconf:type="subtree">
      <te xmlns="urn:ietf:params:xml:ns:yang:ietf-te">
        <tunnels>
          <tunnel>
            <name>Tunnel1</name>
            <identifier/>
            <state>
                <one-way-delay/>
              </te-telemetry>
            </state>
          </tunnel>
        </tunnels>
      </te>
    </filter>
    <period>500</period>
    <encoding>encode-xml</encoding>
  </establish-subscription>
</netconf:rpc>

This example shows the way for a client to subscribe to the telemetry information for all VNs. The telemetry parameter that the client is interested in is one-way-delay and one-way-utilized-bandwidth.

<netconf:rpc netconf:message-id="101"
  xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push:1.0">
    <filter netconf:type="subtree">
        <vn>
          <vn-list>
            <vn-id/>
            <vn-name/>
              <one-way-delay/>
              <one-way-utilized-bandwidth/>
            </vn-telemetry>
          </vn-list>
        </vn>
      </vn-state>
    </filter>
    <period>500</period>
  </establish-subscription>
</netconf:rpc>
6. YANG Data Tree

module: ietf-te-kpi-telemetry
augment /te:te:tunnels/te:tunnel:
  +--rw te-scaling-intent
      +--rw threshold-time?     uint32
      +--rw cooldown-time?      uint32
      +--rw scale-in-operation-type?  scaling-criteria-operation
          +--rw performance-type       identityref
          +--rw threshold-value?      string
          +--rw te-telemetry-tunnel-ref?   -> /te:te/tunnels/tunnel/name
      +--rw scale-out-intent
          +--rw threshold-time?     uint32
          +--rw cooldown-time?      uint32
          +--rw scale-out-operation-type?  scaling-criteria-operation
          +--rw scaling-condition* [performance-type]
              +--rw performance-type       identityref
              +--rw threshold-value?      string
              +--rw te-telemetry-tunnel-ref?   -> /te:te/tunnels/tunnel/name
  +--ro te-telemetry
     +--ro id?            string
     +--ro performance-metrics-one-way
         +--ro one-way-delay?     uint32
         +--ro one-way-delay-normality?  te-types:performance-metrics-normality
         +--ro one-way-residual-bandwidth?
             rt-types:bandwidth-ieee-float32
         +--ro one-way-residual-bandwidth-normality?  te-types:performance-metrics-normality
         +--ro one-way-available-bandwidth?
             rt-types:bandwidth-ieee-float32
         +--ro one-way-available-bandwidth-normality?  te-types:performance-metrics-normality
         +--ro one-way-utilized-bandwidth?
             rt-types:bandwidth-ieee-float32
         +--ro one-way-utilized-bandwidth-normality?  te-types:performance-metrics-normality
     +--ro performance-metrics-two-way
         +--ro two-way-delay?     uint32
         +--ro two-way-delay-normality?  te-types:performance-metrics-normality
         +--ro te-ref?
             -> /te:te/tunnels/tunnel/name
 module: ietf-vn-kpi-telemetry
augment /vn:vn/vn:vn-list:
++--rw vn-scaling-intent
  ++--rw scale-in-intent
    |     ++--rw threshold-time?             uint32
    |     ++--rw cooldown-time?             uint32
    |     ++--rw scale-in-operation-type?   scaling-criteria-operation
    |     ++--rw scaling-condition* [performance-type]
    |         ++--rw performance-type           identityref
    |         ++--rw threshold-value?           string
    |         ++--rw te-telemetry-tunnel-ref?
    |             -> /te:te/tunnels/tunnel/name
  ++--rw scale-out-intent
    |     ++--rw threshold-time?             uint32
    |     ++--rw cooldown-time?              uint32
    |     ++--rw scale-out-operation-type?   scaling-criteria-operation
    |     ++--rw scaling-condition* [performance-type]
    |         ++--rw performance-type           identityref
    |         ++--rw threshold-value?           string
    |         ++--rw te-telemetry-tunnel-ref?
    |             -> /te:te/tunnels/tunnel/name
++--ro vn-telemetry
  ++--ro performance-metrics-one-way
    |     ++--ro one-way-delay?                           uint32
    |     ++--ro one-way-delay-normality?
    |         te-types:performance-metrics-normality
    |     ++--ro one-way-residual-bandwidth?
    |         rt-types:bandwidth-ieee-float32
    |     ++--ro one-way-residual-bandwidth-normality?
    |         te-types:performance-metrics-normality
    |     ++--ro one-way-available-bandwidth?
    |         rt-types:bandwidth-ieee-float32
    |     ++--ro one-way-available-bandwidth-normality?
    |         te-types:performance-metrics-normality
    |     ++--ro one-way-utilized-bandwidth?
    |         rt-types:bandwidth-ieee-float32
    |     ++--ro one-way-utilized-bandwidth-normality?
    |         te-types:performance-metrics-normality
  ++--ro performance-metrics-two-way
    |     ++--ro two-way-delay?             uint32
    |     ++--ro two-way-delay-normality?
    |         te-types:performance-metrics-normality
  ++--ro grouping-operation?            grouping-operation
augment /vn:vn/vn:vn-list/vn:vn-member-list:
  ++--ro vn-member-telemetry
  ++--ro performance-metrics-one-way
    |     ++--ro one-way-delay?                           uint32
    |     ++--ro one-way-delay-normality?
    |         te-types:performance-metrics-normality
    |     ++--ro one-way-residual-bandwidth?
    |         rt-types:bandwidth-ieee-float32
    |     ++--ro one-way-residual-bandwidth-normality?
    |         te-types:performance-metrics-normality
    |     ++--ro one-way-available-bandwidth?
    |         rt-types:bandwidth-ieee-float32
    |     ++--ro one-way-available-bandwidth-normality?
    |         te-types:performance-metrics-normality
7. Yang Data Model

7.1. ietf-te-kpi-telemetry model

The YANG code is as follows:

<CODE BEGINS> file "ietf-te-kpi-telemetry@2019-04-18.yang"

module ietf-te-kpi-telemetry {
  yang-version 1.1;
  prefix te-tel;

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

  /* Note: The RFC Editor will replace YYYY with the number assigned to the RFC once draft-ietf-teas-yang-te becomes an RFC. */

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

  /* Note: The RFC Editor will replace YYYY with the number assigned to the RFC once draft-ietf-teas-yang-te-types becomes an RFC. */

organization
  "IETF Traffic Engineering Architecture and Signaling (TEAS)
   Working Group";
contact
  "Editor: Young Lee <leeyoung@huawei.com>
   Editor: Dhruv Dhody <dhruv.ietf@gmail.com>
   Editor: Ricard Vilalta <ricard.vilalta@cttc.es>
   Editor: Satish Karunanithi <satish.karunanithi@gmail.com>";
description
  "This module describes YANG data model for performance
   monitoring telemetry for te tunnels.

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
(http://trustee.ietf.org/license-info).

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

/* Note: The RFC Editor will replace XXXX with the number
assigned to the RFC once draft-lee-teas-pm-telemetry-
autonomics becomes an RFC.*/

revision 2019-04-18 {
  description
    "Initial revision. This YANG file defines
     a YANG model for TE telemetry.";
  reference "Derived from earlier versions of base YANG files";
}

identity telemetry-param-type {
  description
    "Base identity for telemetry param types";
}

identity one-way-delay {
  base telemetry-param-type;
  description
    "To specify average Delay in one (forward)
     direction";

identity two-way-delay {
    base telemetry-param-type;
    description "To specify average Delay in both (forward and reverse) directions";
RFC7823: Performance-Based Path Selection for Explicitly Routed Label Switched Paths (LSPs) Using TE Metric Extensions";
}

identity one-way-delay-variation {
    base telemetry-param-type;
    description "To specify average Delay Variation in one (forward) direction";
RFC7823: Performance-Based Path Selection for Explicitly Routed Label Switched Paths (LSPs) Using TE Metric Extensions";
}

identity two-way-delay-variation {
    base telemetry-param-type;
    description "To specify average Delay Variation in both (forward and reverse) directions";
RFC7823: Performance-Based Path Selection for Explicitly Routed Label Switched Paths (LSPs) Using TE Metric Extensions";
}

identity utilized-bandwidth {
base telemetry-param-type;
description
"To specify utilized bandwidth over the specified source
and destination.";
reference
"RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
RFC7823: Performance-Based Path Selection for Explicitly
Routed Label Switched Paths (LSPs) Using TE Metric
Extensions";
}

identity utilized-percentage {
  base telemetry-param-type;
description
  "To specify utilization percentage of the entity
   (e.g., tunnel, link, etc.)";
}
typedef scaling-criteria-operation {
type enumeration {
  enum AND {
    description
      "AND operation";
  }
  enum OR {
    description
      "OR operation";
  }
}
description
  "Operations to analyze list of scaling criteria";
}
grouping scaling-duration {
description
  "Base scaling criteria durations";
leaf threshold-time {
  type uint32;
  units "seconds";
  description
    "The duration for which the criteria must hold true";
}
leaf cooldown-time {
  type uint32;
  units "seconds";
  description
    "The duration for which the criteria must hold true";
}
"The duration after a scaling-in/scaling-out action has been triggered, for which there will be no further operation";
}
}
grouping scaling-criteria {
    description
        "Grouping for scaling criteria";
    leaf performance-type {
        type identityref {
            base telemetry-param-type;
        }
        description
            "Reference to the tunnel level telemetry type";
    }
    leaf threshold-value {
        type string;
        description
            "Scaling threshold for the telemetry parameter type";
    }
    leaf te-telemetry-tunnel-ref {
        type leafref {
            path "/te:te/te:tunnels/te:tunnel/te:name";
        }
        description
            "Reference to tunnel";
    }
}
}
grouping scaling-in-intent {
    description
        "Basic scaling in intent";
    uses scaling-duration;
    leaf scale-in-operation-type {
        type scaling-criteria-operation;
        default "AND";
        description
            "Operation to be applied to check between scaling criterias to check if the scale in threshold condition has been met. Defaults to AND";
    }
    list scaling-condition {
        key "performance-type";
        description
            "Scaling conditions";
        uses scaling-criteria;
    }
}
grouping scaling-out-intent {
  description
    "Basic scaling out intent";
  uses scaling-duration;
  leaf scale-out-operation-type {
    type scaling-criteria-operation;
    default "OR";
    description
      "Operation to be applied to check between scaling criterias to check if the scale out threshold condition has been met. Defaults to OR";
  }
  list scaling-condition {
    key "performance-type";
    description
      "Scaling conditions";
    uses scaling-criteria;
  }
}

augment "/te:te/te:tunnels/te:tunnel" {
  description
    "Augmentation parameters for config scaling-criteria TE tunnel topologies. Scale in/out criteria might be used for network autonemics in order the controller to react to a certain set of monitored params.";
  container te-scaling-intent {
    description
      "scaling intent";
    container scale-in-intent {
      description
        "scale-in";
      uses scaling-in-intent;
    }
    container scale-out-intent {
      description
        "scale-out";
      uses scaling-out-intent;
    }
  }
  container te-telemetry {
    config false;
    description

7.2. ietf-vn-kpi-telemetry model

The YANG code is as follows:

```yang
module ietf-vn-kpi-telemetry {
  yang-version 1.1;
  prefix vn-tel;

  import ietf-vn {
    prefix vn;
    reference
      "RFC YYYY: A YANG Data Model for VN Operation";
  }

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

  leaf id {
    type string;
    description
      "Id of telemetry param";
  }

  leaf te-ref {
    type leafref {
      path "/te:te/te:tunnels/te:tunnel/te:name";
    }
    description
      "Reference to measured te tunnel";
  }
}
```
Tunnels and Interfaces};

} /* Note: The RFC Editor will replace YYYY with the number assigned to the RFC once draft-ietf-teas-yang-te becomes an RFC. */

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

/* Note: The RFC Editor will replace YYYY with the number assigned to the RFC once draft-ietf-teas-yang-te-types becomes an RFC. */

import ietf-te-kpi-telemetry {
  prefix te-kpi;
  reference
    "RFC YYYY: YANG models for VN & TE Performance Monitoring Telemetry and Scaling Intent Autonomics";
}

/* Note: The RFC Editor will replace YYYY with the number assigned to the RFC once draft-lee-teas-actn-pm-telemetry-autonomics becomes an RFC. */

organization
  "IETF Traffic Engineering Architecture and Signaling (TEAS) Working Group";
contact
  "Editor: Young Lee <leeyoung@huawei.com>
   Editor: Dhruv Dhody <dhruv.ietf@gmail.com>
   Editor: Ricard Vilalta <ricard.vilalta@cttc.es>
   Editor: Satish Karunanithi <satish.karunanithi@gmail.com>";

description
  "This module describes YANG data models for performance monitoring telemetry for vn."

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
revision 2019-04-18 {
  description
    "Initial revision. This YANG file defines
    the VN telemetry.";
  reference "Derived from earlier versions of base YANG files";
}

typedef grouping-operation {
  type enumeration {
    enum MINIMUM {
      description
        "Select the minimum param";
    }
    enum MAXIMUM {
      description
        "Select the maximum param";
    }
    enum MEAN {
      description
        "Select the MEAN of the params";
    }
    enum STD_DEV {
      description
        "Select the standard deviation of the
        monitored params";
    }
    enum AND {
      description
        "Select the AND of the params";
    }
    enum OR {
      description
        "Select the OR of the params";
    }
  }
  description
}
"Operations to analyze list of monitored params";
}
grouping vn-telemetry-param {
  description
  "augment of te-kpi:telemetry-param for VN specific params";
  leaf-list te-grouped-params {
    type leafref {
      path "/te:te/te:tunnels/te:tunnel/te-kpi:te-telemetry/te-kpi:id";
    }
    description
    "Allows the definition of a vn-telemetry param as a grouping of underlying TE params";
  }
  leaf grouping-operation {
    type grouping-operation;
    description
    "describes the operation to apply to te-grouped-params";
  }
}

augment "/vn:vn/vn:vn-list" {
  description
  "Augmentation parameters for state TE VN topologies.";
  container vn-scaling-intent {
    description
    "scaling intent";
    container scale-in-intent {
      description
      "VN scale-in";
      uses te-kpi:scaling-in-intent;
    }
    container scale-out-intent {
      description
      "VN scale-out";
      uses te-kpi:scaling-out-intent;
    }
  }
  container vn-telemetry {
    config false;
    description
    "VN telemetry params";
    uses te-types:performance-metrics-attributes;
    leaf grouping-operation {
      type grouping-operation;
      description
8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF users to a preconfigured subset of all available NETCONF protocol operations and content. The NETCONF Protocol over Secure Shell (SSH) [RFC6242] describes a method for invoking and running NETCONF within a Secure Shell (SSH) session as an SSH subsystem. The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

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

There are a number of data nodes defined in this YANG module that 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.,
edit-config) to these data nodes without proper protection can have
a negative effect on network operations. These are the subtrees and
data nodes and their sensitivity/vulnerability:

/te:te/te:tunnels/te:tunnel/te-scaling-intent/scale-in-intent
/te:te/te:tunnels/te:tunnel/te-scaling-intent/scale-out-intent

/vn:vn/vn:vn-list/vn-scaling-intent/scale-in-intent
/vn:vn/vn:vn-list/vn-scaling-intent/scale-out-intent

9. IANA Considerations

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

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

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

This document registers the following YANG modules in the YANG
Module.

Names registry [RFC7950]:

-----------------------------------------------
name:     ietf-te-kpi-telemetry
prefix:   te-tel
reference: RFC XXXX (TDB)

10. Acknowledgements

We thank Rakesh Gandhi, Tarek Saad and Igor Bryskin for useful discussions and their suggestions for this work.

11. References

11.1. Normative References


11.2. Informative References


12. Contributors

Authors’ Addresses

Young Lee
Huawei Technologies
5340 Legacy Drive Suite 173
Plano, TX 75024, USA

Email: leeyoung@huawei.com

Dhruv Dhody
Huawei Technology
Leela Palace
Bangalore, Karnataka 560008
India

Email: dhruv.dhody@huawei.com

Satish Karunanithi
Huawei Technology
Leela Palace
Bangalore, Karnataka 560008
India

Email: satish.karunanithi@gmail.com
Ricard Vilalta
Centre Tecnologic de Telecomunicacions de Catalunya (CTTC/CERCA)
Av. Carl Friedrich Gauss 7
08860 - Castelldefels
Barcelona (Spain)
Email: ricard.vilalta@cttc.es

Daniel King
Lancaster University
Email: d.king@lancaster.ac.uk

Daniele Ceccarelli
Ericsson
Torshamnsgatan, 48
Stockholm, Sweden
Email: daniele.ceccarelli@ericsson.com
Interworking of GMPLS Control and Centralized Controller System

draft-zheng-teas-gmpls-controller-inter-work-03

Abstract

Generalized Multi-Protocol Label Switching (GMPLS) control allows each network element (NE) to perform local resource discovery, routing and signaling in a distributed manner.

On the other hand, with the development of software-defined transport networking technology, a set of NEs can be controlled via centralized controller hierarchies to address the issue from multi-domain, multi-vendor and multi-technology. An example of such centralized architecture is ACTN controller hierarchy described in RFC 8453.

Instead of competing with each other, both the distributed and the centralized control plane have their own advantages, and should be complementary in the system. This document describes how the GMPLS distributed control plane can interwork with a centralized controller system in a transport network.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at

This Internet-Draft will expire on August 19, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with
respect to this document. Code Components extracted from this
document must include Simplified BSD License text as described in
Section 4.e of the Trust Legal Provisions and are provided without
warranty as described in the Simplified BSD License.

Conventions used in this document

Table of Contents

1. Introduction ................................................ 3
2. Overview .................................................... 4
2.1. Overview of GMPLS Control Plane ....................... 4
2.2. Overview of Centralized Controller System ............. 4
2.3. GMPLS Control Interwork with Centralized Controller System 4
3. Link Management Protocol .................................. 6
4. Routing Options ........................................... 6
   4.1. OSPF-TE ............................................. 6
   4.2. ISIS-TE ............................................. 6
   4.3. Netconf/RESTconf .................................. 7
5. Path Computation .......................................... 7
   5.1. Constraint-based Path Computing in GMPLS Control .... 7
   5.2. Path Computation Element (PCE) ...................... 7
6. Signaling Options .......................................... 8
   6.1. RSVP-TE ............................................. 8
7. Interworking Scenarios ..................................... 8
1. Introduction

Generalized Multi-Protocol Label Switching (GMPLS) [RFC3945] extends MPLS to support different classes of interfaces and switching capabilities such as Time-Division Multiplex Capable (TDM), Lambda Switch Capable (LSC), and Fiber-Switch Capable (FSC). Each network element (NE) running a GMPLS control plane collects network information from other NEs and supports service provisioning through signaling in a distributed manner. More generic description for Traffic-engineering networking information exchange can be found in [RFC7926].

On the other hand, Software-Defined Networking (SDN) technologies have been introduced to control the transport network in a centralized manner. Central controllers can collect network information from each node and provision services to corresponding nodes. One of the examples is the Abstraction and Control of Traffic Engineered Networks (ACTN) [RFC8453], which defines a hierarchical architecture with Provisioning Network Controller(PNC), Multi-domain Service Coordinator(MDSC) and Customer Network Controller(CNC) as central controllers for different network abstraction levels. A Path Computation Element (PCE) based approach has been proposed as Application-Based Network Operations (ABNO) in [RFC7491].

In such centralized controller architectures, GMPLS can be applied for the NE-level control. A central controller may support GMPLS enabled domains and may interact with a GMPLS enabled domain where the GMPLS control plane does the service provisioning from ingress to egress. In this case the centralized controller sends the request to the ingress node and does not have to configure all NEs along the path through the domain from ingress to egress thus leveraging the GMPLS control plane. This document describes how GMPLS control interworks with centralized controller system in transport network.
2. Overview

In this section, overviews of GMPLS control plane and centralized controller system are discussed as well as the interactions between the GMPLS control plane and centralized controllers.

2.1. Overview of GMPLS Control Plane

GMPLS separates the control plane and the data plane to support time-division, wavelength, and spatial switching, which are significant in transport networks. For the NE level control in GMPLS, each node runs a GMPLS control plane instance. Functionalities such as service provisioning, protection, and restoration can be performed via GMPLS communication among multiple NEs. At the same time, the controller can also collect node and link resources in the network to construct the network topology and compute routing paths for serving service requests.

Several protocols have been designed for GMPLS control [RFC3945] including link management [RFC4204], signaling [RFC3471], and routing [RFC4202] protocols. The controllers applying these protocols communicate with each other to exchange resource information and establish Label Switched Paths (LSPs). In this way, controllers in different nodes in the network have the same view of the network topology and provision services based on local policies.

2.2. Overview of Centralized Controller System

With the development of SDN technologies, a centralized controller architecture has been introduced to transport networks. One example architecture can be found in ACTN [RFC8453]. In such systems, a controller is aware of the network topology and is responsible for provisioning incoming service requests.

Multiple hierarchies of controllers are designed at different levels implementing different functions. This kind of architecture enables multi-vendor, multi-domain, and multi-technology control. For example, an higher-level controller coordinates several lower-level controllers controlling different domains, for topology collection and service provisioning. Vendor-specific features can be abstracted between controllers, and standard API (e.g., generated from RESTconf/YANG) is used.

2.3. GMPLS Control Interwork with Centralized Controller System

Besides the GMPLS and the interactions among the controller hierarchies, it is also necessary for the controllers to communicate with the network elements. Within each domain, GMPLS control can be applied to each NE. The bottom-level central controller can act as a NE to collect network information and initiate LSP. Figure 1 shows
an example of GMPLS interworking with centralized controllers (ACTN terminologies are used in the figure).

![Diagram of GMPLS interworking with controllers]

**Figure 1: Example of GMPLS interworks with Controllers**

In Figure 1, each domain has the GMPLS control plane enabled at the physical network level. The PNC can exploit GMPLS capability implemented in the domain to listen to the IGP routing protocol messages (OSPF LSAs for example) that the GMPLS control plane instances are disseminating into the network and thus learn the network topology. For path computation in the domain with PNC implementing a PCE, PCCs (e.g. NEs, other controller/PCE) use PCEP to ask the PNC for a path and get replies. The MDSC communicates with PNCs using for example REST/RESTConf based on YANG data models. As a PNC has learned its domain topology, it can report the topology to the MDSC. When a service arrives, the MDSC computes the path and coordinates PNCs to establish the corresponding LSP segment.

Alternatively, the NETCONF protocol can be used to retrieve topology information utilizing the e.g. [TE-topo] Yang model and the technology-specific YANG model augmentations required for the
specific network technology. The PNC can retrieve topology information from any NE (the GMPLS control plane instance of each NE in the domain has the same topological view), construct the topology of the domain and export an abstracted view to the MDSC. Based on the topology retrieved from multiple PNCs, the MDSC can create topology graph of the multi-domain network, and can use it for path computation. To setup a service, the MDSC can exploit e.g. [TE-Tunnel] Yang model together with the technology-specific YANG model augmentations.

3. Link Management Protocol

Link management protocol (LMP) [RFC4204] runs between a pair of nodes and is used to manage TE links. In addition to the setup and maintenance of control channels, LMP can be used to verify the data link connectivity and correlate the link property. In this way, link resources, which are fundamental resources in the network, are discovered by both ends of the link.

4. Routing Options

In GMPLS control, link state information is flooded within the network as defined in [RFC4202]. Each node in the network can build the network topology according to the flooded link state information. Routing protocols such as OSPF-TE [RFC4203] and ISIS-TE [RFC5307] have been extended to support different interfaces in GMPLS.

In centralized controller system, central controller can be placed at the GMPLS network and passively receive the information flooded in the network. In this way, the central controller can construct and update the network topology.

4.1. OSPF-TE

OSPF-TE is introduced for TE networks in [RFC3630]. OSPF extensions have been defined in [RFC4203] to enable the capability of link state information for GMPLS network. Based on this work, OSPF protocol has been extended to support technology-specific routing. The routing protocol for OTN, WSON and optical flexi-grid network are defined in [RFC7138], [RFC7688] and [RFC8363], respectively.

4.2. ISIS-TE

ISIS-TE is introduced for TE networks in [RFC5305] and is extended to support GMPLS routing functions [RFC5307], and has been updated to [RFC7074] to support the latest GMPLS switching capability and Types fields.
4.3. Netconf/RESTconf

Netconf [RFC6241] and RESTconf [RFC8040] protocols are originally used for network configuration. Besides, these protocols can also be used for topology retrieval by using topology-related YANG models, such as [RFC8345] and [TE-topo]. These protocols provide a powerful mechanism for notification that permits to notify the client about topology changes.

5. Path Computation

Once a controller learns the network topology, it can utilize the available resources to serve service requests by performing path computation. Due to abstraction, the controllers may not have sufficient information to compute the optimal path. In this case, the controller can interact with other controllers by sending Yang Path Computation requests [PAT-COMP] to compute a set of potential optimal paths and then, based on its own constraints, policy and specific knowledge (e.g. cost of access link) can choose the more feasible path for service e2e path setup.

Path computation is one of the key objectives in various types of controllers. In the given architecture, it is possible for different components that have the capability to compute the path.

5.1. Constraint-based Path Computing in GMPLS Control

In GMPLS control, a routing path is computed by the ingress node [RFC3473] and is based on the ingress node TED. Constraint-based path computation is performed according to the local policy of the ingress node.

5.2. Path Computation Element (PCE)

PCE has been introduced in [RFC4655] as a functional component that provides services to compute path in a network. In [RFC5440], the path computation is accomplished by using the Traffic Engineering Database (TED), which maintains the link resources in the network. The emergence of PCE efficiently improve the quality of network planning and offline computation, but there is a risk that the computed path may be infeasible if there is a diversity requirement, because stateless PCE has no knowledge about the former computed paths.

To address this issue, stateful PCE has been proposed in [RFC8231]. Besides the TED, an additional LSP Database (LSP-DB) is introduced to archive each LSP computed by the PCE. In this way, PCE can easily figure out the relationship between the computing path and former computed paths. In this approach, PCE provides computed paths to
PCC, and then PCC decides which path is deployed and when to be established.

In PCE Initiation [RFC8281], PCE is allowed to trigger the PCC to setup, maintenance, and teardown of the PCE-initiated LSP under the stateful PCE model. This would allow a dynamic network that is centrally controlled and deployed.

In centralized controller system, the PCE can be implemented in a central controller, and the central controller performs path computation according to its local policies. On the other hand, the PCE can also be placed outside of the central controller. In this case, the central controller acts as a PCC to request path computation to the PCE through PCEP. One of the reference architecture can be found at [RFC7491].

6. Signaling Options

Signaling mechanisms are used to setup LSPs in GMPLS control. Messages are sent hop by hop between the ingress node and the egress node of the LSP to allocate labels. Once the labels are allocated along the path, the LSP setup is accomplished. Signaling protocols such as RSVP-TE [RFC3473] have been extended to support different interfaces in GMPLS.

6.1. RSVP-TE

RSVP-TE is introduced in [RFC3209] and extended to support GMPLS signaling in [RFC3473]. Several label formats are defined for a generalized label request, a generalized label, suggested label and label sets. Based on [RFC3473], RSVP-TE has been extended to support technology-specific signaling. The RSVP-TE extensions for OTN, WSON, optical flexi-grid network are defined in [RFC7139], [RFC7689], and [RFC7792], respectively.

7. Interworking Scenarios

7.1. Topology Collection & Synchronization

Topology information is necessary on both network elements and controllers. The topology on network element is usually raw information, while the topology on the controller can be either raw or abstracted. Three different abstraction methods have been described in [RFC8453], and different controllers can select the corresponding method depending on application.

When there are changes in the network topology, the impacted network element(s) need to report changes to all the other network elements, together with the controller, to sync up the topology information. The inter-NE synchronization can be achieved via protocols mentioned
The topology synchronization between NEs and controllers can either be achieved by routing protocols OSPF-TE/PCEP-LS in [PCEP-LS] or Netconf protocol notifications with YANG model.

7.2. Multi-domain/layer Service Provisioning

Based on the topology information on controllers and network elements, service provisioning can be deployed. Plenty of methods have been specified for single domain service provisioning, such as using PCEP and RSVP-TE.

Multi-domain/layer service provisioning would request coordination among the controller hierarchies. Given the service request, the end-to-end delivery procedure may include interactions at any level (i.e. interface) in the hierarchy of the controllers (e.g. MPI and SBI for ACTN). The computation for a cross-domain/layer path is usually completed by controllers who have a global view of the topologies. Then the configuration is decomposed into lower layer controllers, to configure the network elements to set up the path.

A combination of the centralized and distributed protocols may be necessary for the interaction between network elements and controller. A typical example would be the PCE Initiation scenario, in which a PCE message (PCInitiate) is sent from the controller to the first-end node, and then trigger a RSVP procedure along the path. Similarly, the interaction between the controller and the ingress node of a domain can be achieved by Netconf protocol with corresponding YANG models, and then completed by running RSVP among the network elements.

7.3. Recovery

The GMPLS recovery functions are described in [RFC4426]. Two models, span protection and end-to-end protection and restoration, are discussed with different protection schemes and message exchange requirements. Related RSVP-TE extensions to support end-to-end recovery is described in [RFC4872]. The extensions in [RFC4872] include protection, restoration, preemption, and rerouting mechanisms for an end-to-end LSP. Besides end-to-end recovery, a GMPLS segment recovery mechanism is defined in [RFC4873]. By introducing secondary record route objects, LSP segment can be switched to another path like fast reroute [RFC4090].

For the recovery with controllers, timely interaction between controller and network elements are required. Usually the re-routing can be decomposed into path computation and delivery, the controller can take some advantage in the path computation due to the global topology view. And the delivery can be achieved by the procedure described in section 7.2.
7.4. Controller Reliability

Given the important role in the network, the reliability of controller is critical. Once a controller is shut down, the network should operate as well. It can be either achieved by controller back up or functionality back up. There are several of controller backup or federation mechanisms in the literature. It is also more reliable to have some function back up in the network element, to guarantee the performance in the network.

8. Manageability Considerations

Each entity in the network, including both controllers and network elements, should be managed properly as it will interact with other entities. The manageability considerations in controller hierarchies and network elements still apply respectively. For the protocols applied in the network, manageability is also requested.

The responsibility of each entity should be clarified. The control of function and policy among different controllers should be consistent via proper negotiation process.

9. Security Considerations

This document provides the interwork between the GMPLS and controller hierarchies. The security requirements in both system still applies respectively. Protocols referenced in this document also have various security considerations, which is also expected to be satisfied.

Other considerations on the interface between the controller and the network element are also important. Such security includes the functions to authenticate and authorize the control access to the controller from multiple network elements. Security mechanisms on the controller are also required to safeguard the underlying network elements against attacks on the control plane and/or unauthorized usage of data transport resources.

10. IANA Considerations

This document requires no IANA actions.

11. References

11.1. Normative References


Zheng et. al Expires August 2019
11.2. Informative References


12. Authors’ Addresses

Haomian Zheng
Huawei Technologies
F3 R&D Center, Huawei Industrial Base,
Bantian, Longgang District,
Shenzhen 518129 P.R.China
Email: zhenghaomian@huawei.com

Xianlong Luo
Huawei Technologies
F3 R&D Center, Huawei Industrial Base,
Bantian, Longgang District,
Shenzhen 518129 P.R.China
Email: luoxianlong@huawei.com

Yunbin Xu
CAICT
Email: xuyunbin@ritt.cn

Yang Zhao
China Mobile
Email: zhaoyangyjy@chinamobile.com

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

Dieter Beller
Nokia
Email: Dieter.Beller@nokia.com
Abstract

This document specifies the optimization in RSVP-TE P2MP tunnel signaling over Resilient MPLS Rings (RMR).

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119.

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 August 2, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect
1. Introduction

Traditional RSVP-TE P2MP tunnel signaling could be quite involving. With RMR, this could be significantly simplified:

There is no need for ERO/RRO/SERO/SRRO or hop by hop routing. The tunnel ingress simply sends PATH messages in one or both directions of the ring, depending on how leaves are best reached. The <S2L Sub-LSP Descriptor List> only needs to list the tunnel leaves, and a transit router does not need to "branch" a PATH message into multiple ones. Therefore, unless there are many tunnel leaves on a huge ring, a single PATH message is enough. In the rare situation of a large tunnel with many leaves to list, a small number of PATH messages should suffice. Additionally, there is no need to signal and maintain individual sub-LSPs (one for each leaf) any more. As a result, corresponding PATH/RESV state is also reduced. Each node only needs to maintain a single PATH state and a single RESV state for each P2MP tunnel, and the RESV state does not need to track individual leaves - it just need to track if a RESV is received from downstream and/or if this node itself is a leaf.

A RESV message is triggered to the PHOP when the RESV state is first created (either because the node is a leaf or because a RESV message is received from downstream) and it is refreshed periodically. A RESV Tear is sent when the RESV state is deleted (when the node is no longer a Leaf and the RESV from downstream has timed out or a RESV Tear is received).

Optionally, the tunnel ingress may not need to list any/all leaves. It could simply send the PATH message around the ring, with the <S2L
Sub-LSP Descriptor List> listing the root itself. Through methods outside the scope of this document, a node determines if it is a leaf of the tunnel, and if yes, it will send back a RESV message. With this, a single PATH message is surely enough.

In this document, leaves in <S2L Sub-LSP Descriptor List> are referred to as explicit leaves, and leaves not listed there but self-determined by ring nodes are referred to as implicit leaves. There could be both explicit and implicit leaves for a tunnel. The ingress allows implicit leaves by including itself as the last one in the <S2L Sub-LSP Descriptor List>.

Optionally, the RESV message could also include a <S2L Sub-LSP Descriptor List> to list all the leaves on the established tunnel so that the each node knows its downstream leaves. In that case, when the set of downstream leaves changes, a RESV message with the new <S2L Sub-LSP Descriptor List> is triggered.

Adding/removing explicit leaves is straightforward. The ingress simply sends a triggered PATH message with new <S2L Sub-LSP Descriptor List>. As it passes around the ring, each node determines if it is an explicit leaf and updates its state accordingly. The triggered PATH message does not have to go all the way to the last leaf - if on a node the <S2L Sub-LSP Descriptor List> in the would-be-sent PATH message is the same as what was sent before, the triggered PATH message will not be sent further.

To indicate that the tunnel signaling is with above mentioned RMR optimizations, a new object is included in the PATH message to specify the Ring ID and direction.

Link/Node protection is achieved by tunneling packets to the next node using the Ring LSP to that node in the other direction. This does not need any additional signaling but is based on a reasonable premise that unicast Ring LSPs are always in place. Once the ingress learns the failure (through IGP discovery or through other error detection/notification mechanisms), global repair kicks in to reach some leaves via PATH message sent in the other direction. Before global repair is finished, traffic continues to flow in the original path except that at the failure point it is tunneled to the next node.

If an RMR is just part of a general RSVP network the optimization can also be applied on the ring nodes. If the tunnel ingress knows the leaves that are on the ring, it could put all those leaves in the single PATH message and construct the ERO/SERO only towards the entry point on the ring. The entry point then includes the RMR object in the PATH messages that it sends. For leaves beyond the ring, the
ingress may include the exit points on the ring as loose hops in the ERO/SERO, and when a ring node needs to send the PATH message off the ring, it removes the RMR object. Details will be provided in future revisions of this document.

2. Specification

2.1. RMR Object

The RMR object is a new object of the following:

- Class Name: RMR
- Class-Num: TBA1 (to be assigned by IANA)
- C-Type: TBA2 (to be assigned by IANA)

The format of the object content following the common object header is the following:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Ring ID (4 octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|D| Flags       |      Reserved                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Following the 4-octect Ring ID, there is an 8-bit Flags field. The first bit of the Flags field indicates the direction. If it is set, it is clockwise direction. Otherwise, it is anti-clockwise.

2.2. Procedures

This section describes the differences in the procedures for ring nodes to set up RSVP-TE P2MP tunnels across the ring, compared to the conventional non-RMR-aware case. For now it is assumed that all nodes (ingress, transit, and leaves) on the tunnel are on the ring.

More details will be provided in future revisions.

2.2.1. PATH Message/State

The tunnel ingress includes the RMR object with the Ring ID and the direction flag bit set accordingly. The explicit tunnel leaves are encoded in the <S2L Sub-LSP Descriptor List>, and no ERO/SERO is included. If the tunnel allows implicit leaves, the descriptor list encodes the ingress itself as the last element. The message is sent...
to the next node on the ring in the direction specified in the RMR
descriptor list, w/o using ERO/SERO or hop-by-hop routing.

When a node receives a PATH message with the RMR object, it checks if
itself is listed in the <S2L Sub-LSP Descriptor List>, or if the <S2L
Sub-LSP Descriptor List> encodes the tunnel ingress as the last
element and this node itself is an implicit leaf. If yes, it creates
corresponding RESV state and sends a RESV message to the PHOP.

The receiving node removes itself from the <S2L Sub-LSP Descriptor
List> in the PATH message, and saves the list locally. The PATH
message is sent to the next node on the ring in the specified
direction with the saved <S2L Sub-LSP Descriptor List>, if one of the
following conditions is met:

- The <S2L Sub-LSP Descriptor List> encodes the tunnel ingress
  itself as the last element.

- The <S2L Sub-LSP Descriptor List> is not empty and either the PATH
  state is newly created or the <S2L Sub-LSP Descriptor List> is
different from the previously saved one.

If <S2L Sub-LSP Descriptor List> is empty and different from the
previously saved one, a PATH Teardown is sent instead.

2.2.2. RESV Message/State

A ring node may know that it is a leaf when the PATH message is first
processed as described in the previous section. In case of implicit
leaves, it may become a leaf after the PATH messages has been
processed. A non-leaf node may also receive a RESV message from its
NHOP. In all these cases, the node creates RESV state and sends a
RESV message to the PHOP, w/o encoding RRO/SRRO.

If a ring node was a leaf but stops being a leaf, either because it
is no longer listed in the <S2L Sub-LSP Descriptor List> or it is no
longer an implicit leaf, it removes/updates corresponding local
state. A RESV Teardown is sent to the PHOP if there is no RESV
received from its downstream either.

3. Security Considerations

This document does not introduce new security risks.
4. Acknowledgements

5. Normative References

[I-D.ietf-mpls-rmr]
Kompella, K. and L. Contreras, "Resilient MPLS Rings",

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,

Yasukawa, Ed., "Extensions to Resource Reservation
Protocol - Traffic Engineering (RSVP-TE) for Point-to-
Multipoint TE Label Switched Paths (LSPs)!", RFC 4875,
DOI 10.17487/RFC4875, May 2007,

Authors’ Addresses

Zhaohui Zhang
Juniper Networks
EMail: zzhang@juniper.net

Abhishek Deshmukh
Juniper Networks
EMail: adeshmukh@juniper.net

Ravi Singh
Juniper Networks
EMail: ravis@juniper.net