YANG models for VN & TE Performance Monitoring Telemetry and Scaling Intent Autonomics

draft-ietf-teas-actn-pm-telemetry-autonomics-00

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 January 1, 2020.

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.

Table of Contents

1. Introduction ................................................... 3
   1.1. Terminology ............................................... 4
   1.2. Tree diagram .............................................. 5
   1.3. Prefixes in Data Node Names ............................... 5
2. Use-Cases ...................................................... 5
3. Design of the Data Models ....................................... 7
   3.1. TE KPI Telemetry Model .................................... 7
   3.2. VN KPI Telemetry Model .................................... 8
4. Autonomic Scaling Intent Mechanism ............................. 9
5. Notification .................................................. 11
   5.1. YANG Push Subscription Examples .......................... 11
6. YANG Data Tree ................................................ 13
7. YANG Data Model ............................................... 15
   7.1. ietf-te-kpi-telemetry model ............................... 15
   7.2. ietf-vn-kpi-telemetry model ............................... 21
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 these 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 inter-exchangeable 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.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>rt</td>
<td>ietf-routing-types</td>
<td>[RFC8294]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[TE-Tunnel]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[TE-Types]</td>
</tr>
<tr>
<td>te-tel</td>
<td>ietf-te-kpi-telemetry</td>
<td>[This I-D]</td>
</tr>
<tr>
<td>vn</td>
<td>ietf-vn</td>
<td>[VN]</td>
</tr>
<tr>
<td>vn-tel</td>
<td>ietf-vn-kpi-telemetry</td>
<td>[This I-D]</td>
</tr>
</tbody>
</table>

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.

+------------+          +--------------+
|  TE-Tunnel |          |    TE KPI    |
|   Model    |<---------|  Telemetry   |
+------------+ augments |     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 \( \text{delay\_vn\_member\_1, delay\_vn\_member\_2,.. delay\_vn\_member\_N} \). 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:
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>
<te-telemetry xmlns="urn:ietf:params:xml:ns:yang:
ietf-te-kpi-telemetry">
<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-state xmlns="urn:ietf:params:xml:ns:yang:
ietf-vn">
<vn>
<vn-list>
<vn-id/>
<vn-name/>
<vn-telemetry xmlns="urn:ietf:params:xml:ns:yang:
ietf-vn-kpi-telemetry">
<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 scale-in-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
  +++rw te-scaling-out-intent
  +++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 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

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:

```flex
<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.*/

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";
    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 one-way-delay-variation {
    base telemetry-param-type;
    description
        "To specify average Delay Variation in one (forward) direction";
    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 two-way-delay-variation {
    base telemetry-param-type;
    description
        "To specify average Delay Variation in both (forward and reverse)
        directions";
    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-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 analize list of scaling criterias";
}

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 cool down";
}
"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 autonomies 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
    "This section outlines the telemetry parameters
    related to scaling in an autonomous network.
    These parameters provide valuable insights into
    the current state and capacity of the network.
    This information is crucial for making informed
    decisions about scaling strategies and capacity
    planning.
    The telemetry parameters include:
    - Performance Metrics: Essential for evaluating
      the performance of the network.
      Monitoring these metrics helps identify
      bottlenecks and areas for improvement.
    - Utilization Levels: Tracking the utilization
      levels allows for efficient resource allocation.
      This is particularly important when scaling
      in order to avoid overprovisioning or
      underutilization.
    - Traffic Load: Understanding the traffic load
      aids in forecasting future needs and
      planning capacity expansions.
    - Resource Availability: Ensuring that the
      necessary resources are available for
      successful scaling operations.
    - Performance Indicators: Key performance
      indicators (KPIs) provide a holistic view
      of the network's performance.
      Monitoring these KPIs helps
      in aligning scaling efforts with
      business objectives.
    - Segmentation Levels: Enabling effective
      segmentation helps in managing
      different services and user groups
      efficiently.
    - Security Parameters: Security parameters
      are critical for maintaining data
      confidentiality and integrity.
    - Quality of Service (QoS): QoS settings
      ensure that different service levels
      are maintained during scaling operations.
    - Service Level Agreements (SLAs):
      SLAs define the expected performance
      levels and service availability.
    - Policy Management: Effective policy
      management is essential for controlling
      and configuring network behavior.
    - Access Control: Access control measures
      ensure that only authorized users
      have access to network resources.
    - Resource Configuration: Configuring
      network resources is crucial for
      ensuring smooth scaling operations.
    - Monitoring and Reporting: Comprehensive
      monitoring and reporting tools
      provide insights into network
      performance and help
      in decision-making.
    - Integration Tools: Integration tools
      facilitate the interoperability
      of different network
      components.
    - Deployment Strategies: Deployment
      strategies determine
      how scaling operations are
      executed.
    - Change Management: Effective change
      management practices
      help in minimizing
      service disruption.
    - Capacity Planning: Accurate
      capacity planning
      is essential for ensuring
      that the network is
      adequately prepared
      to handle future load.
    - Technology Selection: Choosing the
      right technology
      is crucial for
      achieving successful
      scaling outcomes.
    - Regulatory Compliance: Compliance
      with relevant
      regulations and standards
      is necessary for
      successful scaling operations.
    - Scalability Assessment: Regular
      assessment of
      scaling capabilities
      helps in identifying
      areas for improvement.
    - Monitoring Capabilities: Robust
      monitoring capabilities
      are essential for
      identifying issues
      and making informed
      decisions.
    - Support Services: Quality
      support services
      ensure that scaling
      operations proceed
      smoothly.
    - Performance Optimization: Continuous
      optimization of performance
      metrics helps
      in achieving
      increased efficiency.
"telemetry params";
leaf id {
    type string;
    description
    "Id of telemetry param";
}

uses te-types:performance-metrics-attributes;
leaf te-ref {
    type leafref {
        path "/te:te/te:tunnels/te:tunnel/te:name";
    }
    description
    "Reference to measured te tunnel";
}

<CODE ENDS>

7.2. ietf-vn-kpi-telemetry model

The YANG code is as follows:

<CODE BEGINS> file "ietf-vn-kpi-telemetry@2019-04-18.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";
    }

    /* Note: The RFC Editor will replace YYYY with the number
    assigned to the RFC once draft-ietf-teas-actn-vn-yang
    becomes an RFC.*/

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

Tunnels and Interfaces";}

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

import ietf-te-kpi-telemetry {
  prefix te-kpi;
  reference
    "RFC YYYY: YANG models for VN & TE Performance Monitoring
     Telemetry and Scaling Intent Autonomics";
}

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

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-tes-pm-telemetry-autonomics becomes an RFC.*/

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
Futurewei Technologies
5340 Legacy Drive Suite 173
Plano, TX 75024, USA

Email: younglee.tx@gmail.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
Torshammsgatan,48
Stockholm, Sweden
Email: daniele.ceccarelli@ericsson.com
A Yang Data Model for VN Operation

draft-ietf-teas-actn-vn-yang-06

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 January 5, 2020.
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.

Table of Contents

Introduction.....................................................3
1.................................................................3
   1.1. Terminology...............................................4
   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.

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

Table 1: Prefixes and corresponding YANG modules

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

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

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

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

Figure 1. ACTN CMI

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

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:
VN-Member 1       L1-L4
VN-Member 2       L1-L7
VN-Member 3       L2-L4
VN-Member 4       L3-L8

Where L1, L2, L3, L4, L7 and L8 correspond to 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 January 2020
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 ------ | AN 1 | L4 ------ |
| L2 ------ |     | 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 | | MDSC | | BMC | | BM | | BMC | | BM |
+---------------+  +---------------+  +---------------+  +---------------+  +---------------+  +---------------+  +---------------+  +---------------+
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 January 2020 [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.

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 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 /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 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?
   --> /ap/access-point-list/vn-ap/vn-ap-id
      | +--rw multi-src? boolean {multi-src-dest}?
      | +--rw dest
      | +--rw dest?
   --> /ap/access-point-list/vn-ap/vn-ap-id
      | +--rw dest-vn-ap-id?
   --> /ap/access-point-list/vn-ap/vn-ap-id
      | +--rw multi-dest? boolean {multi-src-dest}?
      | +--rw connectivity-matrix-id?
      | +--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 autonamics 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-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?    
  |  -> /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}?
  |  |  |  ++rw connectivity-matrix-id?
  -> /nw:networks/network/node/tet:te/te-node-attribute
     /connectivity-matrices/connectivity-matrix/id
     |  ++ro oper-status?      identityref
     |  ++ro if-selected?      boolean {multi-src-dest}?
     |  ++ro 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?   
     |  -> /ap/access-point-list/vn-ap/vn-ap-id
     |  |  |  --w multi-src?      boolean {multi-src-dest}?
     |  |  |  --w dest           
     |  |  |  |  --w dest?        
     |  -> /ap/access-point-list/access-point-id
     |  |  |  --w dest-vn-ap-id? 
     |  -> /ap/access-point-list/vn-ap/vn-ap-id
     |  |  |  |  --w multi-dest?    boolean {multi-src-dest}?
     |  |  |  |  --w dest         
     |  |  |  |  |  --w dest?     
     |  -> /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 2019-06-20 {
    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-computatione-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 {

Lee & Dhody Expires January 2020 [Page 19]
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
"VN YANG Model                     July 2019";

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

/*add details and any other properties of AP, not associated by a VN CE port, PE port etc. */

/*add details and any other properties of AP, not associated by a VN CE port, PE port etc. */
path "/ap/access-point-list/vn-ap/vn-ap-id";
}

description
"reference to source VNAP";
}
leaf multi-src {
  if-feature multi-src-dest;
  type boolean;
  description
  "Is source part of multi-source, where only one of the source is enabled";
}
}

container dest {

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

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

/*

grouping vn-policy {
    description
    "policy for VN-level 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";
    }
}
*/
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;
}

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";
}

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;
}
}

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.";
    }
}
/*
container multi-src-dest{
    if-feature multi-src-dest;
    description
        "";
}*/
"The selected VN Member when multi-src
and/or mult-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-src] and VN Member 204 (L2 to L4)/304 (L3 to L4) [multi-
  dest] usecase. The selected VN-member is known via the field "if-
  selected" and the corresponding connectivity-matrix-id.
Note that the VN YANG model also include the AP and VNAP which shows various VN using the same AP.

7.1. VN JSON

```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": 0,
                        "abstract-node": "D3",
                        "ltp": "1-0-1"
                    }
                ]
            }
        ]
    }
}
```
{ "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,
"
{"vn-member-id": 204,
"src": {
   "src": 202,
   "src-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,
"src": {  
  "src": 202,  
  "multi-src": true,  
},  
"dest": {  
  "dest": 440,  
},  
"vn-member-id": 304,  
"src": {  
  "src": 303,  
  "src-vn-ap-id": 30303,  
  "multi-src": true,  
  "multi-vn-ap-id": true,  
},  
"dest": {  
  "dest": 440,  
  "src-vn-ap-id": 44003,  
},  
"connectivity-matrix-id": 304,  
"if-selected":true,  
},  
}),  
}]  
},  
}  
}  
}  
}

7.2. TE-topology 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": [  
  
}
<table>
<thead>
<tr>
<th>tp-id</th>
<th>te-tp-id</th>
<th>te</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-0-1</td>
<td>10001</td>
<td></td>
</tr>
<tr>
<td>1-1-0</td>
<td>10100</td>
<td></td>
</tr>
<tr>
<td>2-0-2</td>
<td>20002</td>
<td></td>
</tr>
<tr>
<td>2-2-0</td>
<td>20200</td>
<td></td>
</tr>
</tbody>
</table>
"tp-id": "3-0-3",
"te-tp-id": 30003,
"te": {
   "interface-switching-capability": [ 
       { 
           "switching-capability": "switching-otn",
           "encoding": "lsp-encoding-oduk"
       }
   ]
},
{
   "tp-id": "3-3-0",
   "te-tp-id": 30300,
   "te": {
       "interface-switching-capability": [ 
           { 
               "switching-capability": "switching-otn",
               "encoding": "lsp-encoding-oduk"
           }
       ]
   },
},
{
   "tp-id": "4-0-4",
   "te-tp-id": 40004,
   "te": {
       "interface-switching-capability": [ 
           { 
               "switching-capability": "switching-otn",
               "encoding": "lsp-encoding-oduk"
           }
       ]
   },
},
{
   "tp-id": "4-4-0",
   "te-tp-id": 40400,
   "te": {
       "interface-switching-capability": [ 
           { 
               "switching-capability": "switching-otn",
               "encoding": "lsp-encoding-oduk"
           }
       ]
   },
},
{
   "tp-id": "5-0-5",
}
"te-tp-id": 50005,
"te": {
  "interface-switching-capability": [
    { "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},

{ "tp-id": "5-5-0",
  "te-tp-id": 50500,
  "te": {
    "interface-switching-capability": [
      { "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},

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

{ "tp-id": "6-6-0",
  "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": {  

"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": {
                                "enables": true
                            }
                        }
                    }
                }
            ]
        }
    ]
}
"path-element-id": 1,
  "index": 1,
  "numbered-hop": {
    "address": "4.4.4.4",
    "hop-type": "STRICT"
  }
},

{ "path-element-id": 2,
  "index": 2,
  "numbered-hop": {
    "address": "7.7.7.7",
    "hop-type": "STRICT"
  }
}]
},

"termination-point": [
  {
    "tp-id": "1-0-1",
    "te-tp-id": 10001,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-1-0",
    "te-tp-id": 10100,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {

"tp-id": "2-0-2",
"te-tp-id": 20002,
"te": {
 "interface-switching-capability": [
   { 
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk"
   }
 ]
 },
 "tp-id": "2-2-0",
"te-tp-id": 20200,
"te": {
 "interface-switching-capability": [
   { 
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk"
   }
 ]
 }
},
 "tp-id": "3-0-3",
"te-tp-id": 30003,
"te": {
 "interface-switching-capability": [
   { 
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk"
   }
 ]
 }
},
 "tp-id": "3-3-0",
"te-tp-id": 30300,
"te": {
 "interface-switching-capability": [
   { 
    "switching-capability": "switching-otn",
    "encoding": "lsp-encoding-oduk"
   }
 ]
 }
},
 "tp-id": "4-0-4",
"te-tp-id": 40004,
"te-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"
      }
    ]
  }
}
"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"
                }
            ]
        }
    }
]
{
    "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": "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"
  }
 ]
}
}
"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 January 2020 [Page 52]
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

Lee & Dhody Expires January 2020
Interworking of GMPLS Control and Centralized Controller System

draft-ietf-teas-gmpls-controller-inter-work-01

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 http://www.ietf.org/shadow.html.

This Internet-Draft will expire on January 8, 2020.

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
4. Routing Options................................................6
4.1. OSPF-TE...................................................6
4.2. ISIS-TE...................................................7
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, a 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).

Figure 1: Example of GMPLS/non-GMPLS interworks with Controllers

Figure 1 shows the scenario with two GMPLS domains and one non-GMPLS domain. This system supports the interworking among non-GMPLS domain, GMPLS domain and the controller hierarchies. For domain 1, the network element were not enabled with GMPLS so the control can be purely from the controller, via Netconf/YANG and/or PCEP. For domain 2 and 3, 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. Discovery Options

In GMPLS control, the link connectivity need to be verified between each pair of nodes. In this way, link resources, which are fundamental resources in the network, are discovered by both ends of the link.

3.1. LMP

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.

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 end-to-end 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 in section 3 and 4. 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 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 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 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. Several methods can be used to create the inter-domain path:

1) One end-to-end RSVP-TE session:

In this method, an end-to-end RSVP-TE session from the source NE to the destination NE will be used to create the inter-domain path. 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.
This method requires the interworking of RSVP-TE protocols between different domains.

2) Multiple RSVP-TE sessions for multiple path segments:

In this method, each SDN controller is responsible to create the path segment within its domain.

Note that path segments in the source domain and the destination domain are "asymmetrical" segments, because the configuration of client signal mapping into server layer tunnel is needed at only one end of the segment, while configuration of server layer cross-connect is needed at the other end of the segment. For example, the path segment 1 and 3 in Figure 3 are asymmetrical segments, because one end of the segment requires mapping GE into ODU0, while the other end of the segment requires setting up ODU0 cross-connect.

![Diagram of asymmetrical path segment](image)

Figure 3: Example of asymmetrical path segment

The PCEP / GMPLS protocols should support creation of such asymmetrical segment.

Note also that mechanisms to assign the labels in the inter-domain links are also needed to be considered. There are two possible methods:

2.1) Inter-domain labels assigned by NEs:

The concept of Stitching Label that allows stitching local path segments was introduced in [RFC5150] and [sPCE-ID], in order to form the inter-domain path crossing several different domains. It also describes the BRPC and H-PCE PCInitiate procedure, i.e., the ingress border node of each downstream domain assigns the stitching label for the inter-domain link between the downstream domain and its
upstream neighbor domain, and this stitching label will be passed to
the upstream neighbor domain by PCE protocol, which will be used for
the path segment creation in the upstream neighbor domain.

2.2) Inter-domain labels assigned by SDN controller:

If the resource of inter-domain links are managed by the multi-
domain SDN controller, each single-domain SDN controller can provide
to the multi-domain SDN controller the list of available labels
(e.g. timeslots if OTN is the scenario) using IETF Topology model
and related technology specific extension. Once that multi-domain
SDN controller has computed e2e path RSVP-TE or PCEP can be used in
the different domains to setup related segment tunnel consisting
with label inter-domain information, e.g. for PCEP the label ERO can
be included in the PCInitiate message to indicate the inter-domain
labels, so that each border node of each domain can configure the
correct cross-connect within itself.

7.3. Multi-layer Service Provisioning

For further study. Plan to be updated in the next version.

7.4. 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.

7.5. 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


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

ITU-T Recommendation G.808.3 [G808.3] defines the generic aspects of a Shared Mesh Protection (SMP) mechanism, where the difference between SMP and Shared Mesh Restoration (SMR) is also identified. ITU-T Recommendation G.873.3 [G873.3] defines the protection switching operation and associated protocol for SMP at the Optical Data Unit (ODU) layer. RFC 7412 [RFC7412] provides requirements for any mechanism that would be used to implement SMP in a Multi-Protocol Label Switching - Transport Profile (MPLS-TP) network.

This document updates RFC 4872 [RFC4872] to provide the extensions to the Generalized Multi-Protocol Label Switching (GMPLS) signaling to support the control of the shared mesh protection.

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 January 6, 2020.
# 1. Introduction

RFC 4872 [RFC4872] defines extension of Resource Reservation Protocol - Traffic Engineering (RSVP-TE) to support Shared Mesh Restoration (SMR) mechanism. SMR can be seen as a particular case of pre-planned Label Switched Path (LSP) rerouting that reduces the recovery resource requirements by allowing multiple protecting LSPs to share common link and node resources. The recovery resources for the protecting LSPs are pre-reserved during the provisioning phase, and an explicit restoration signaling is required to activate (i.e., commit resource allocation at the data plane) a specific protecting LSP instantiated during the provisioning phase.
ITU-T Recommendation G.808.3 [G808.3] defines the generic aspects of a shared mesh protection (SMP) mechanism. ITU-T Recommendation G.873.3 [G873.3] defines the protection switching operation and associated protocol for SMP at the Optical Data Unit (ODU) layer. RFC 7412 [RFC7412] provides requirements for any mechanism that would be used to implement SMP in a Multi-Protocol Label Switching - Transport Profile (MPLS-TP) network.

SMP differs from SMR in the activation/protection switching operation. The former activates a protecting LSP via the automatic protection switching (APS) protocol in the data plane when the working LSP fails, while the latter does it via the control plane signaling. It is therefore necessary to distinguish SMP from SMR during provisioning so that each node involved behaves appropriately in the recovery phase when activation of a protecting LSP is done.

This document updates [RFC4872] to provide the extensions to the Generalized Multi-Protocol Label Switching (GMPLS) signaling to support the control of the SMP mechanism. Only the generic aspects for signaling SMP are addressed by this document. The technology-specific aspects are expected to be addressed by other drafts.

2. Conventions Used in This Document

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.

In addition, the reader is assumed to be familiar with the terminology used in [RFC4872] and RFC 4426 [RFC4426].

3. SMP Definition

[G808.3] defines the generic aspects of a SMP mechanism. [G873.3] defines the protection switching operation and associated protocol for SMP at the ODU layer. [RFC7412] provides requirements for any mechanism that would be used to implement SMP in a MPLS-TP network.

The SMP mechanism is based on pre-computed protection transport entities that are pre-configured into the network elements. Preconfiguration here means pre-reserving resources for the protecting LSPs without activating a particular protecting LSP (e.g. in circuit networks, the cross-connects in the intermediate nodes of the protecting LSP are not pre-established). Pre-configuring but not activating the protecting LSP allows the common link and node resources in a protecting LSP to be shared by multiple working LSPs.
that are physically (i.e., link, node, Shared Risk Link Group (SRLG), etc.) disjoint. Protecting LSPs are activated in response to failures of working LSPs or operator’s commands by means of the APS protocol that operates in the data plane. The APS protocol messages are exchanged along the protecting LSP. SMP is always revertive.

SMP has a lot of similarity to SMR except that the activation in case of SMR is achieved by control plan signaling during the recovery operation while SMP is done by APS protocol in the data plane. SMP has advantages with regard to the recovery speed compared with SMR.

4. GMPLS Signaling Extension for SMP

Consider the following network topology:

```
A---B---C---D
  \   /  
  E---F---G
    /   /  
H---I---J---K
```

Figure 1: An example of SMP topology

The working LSPs [A,B,C,D] and [H,I,J,K] could be protected by [A,E,F,G,D] and [H,E,F,G,K], respectively. Per RFC 3209 [RFC3209], in order to achieve resource sharing during the signaling of these protecting LSPs, they must have the same Tunnel Endpoint Address (as part of their SESSION object). However, these addresses are not the same in this example. Similar to SMR, a new LSP Protection Type of the secondary LSP is defined as "Shared Mesh Protection" (see PROTECTION object defined in [RFC4872]) to allow resource sharing along nodes E, F, and G. In this case, the protecting LSPs are not merged (which is useful since the paths diverge at G), but the resources along E, F, G can be shared.

When a failure, such as Signal Fail (SF) and Signal Degrade (SD), occurs on one of the working LSPs (say working LSP [A,B,C,D]), the end-node (say node A) that detects the failure initiates the protection switching operation. The end-node A will send a protection switching request APS message (for example SF) to its adjacent (downstream) intermediate node (say node E) to activate setting up the corresponding protecting LSP and will wait for a confirmation message from node E. If the protection resource is available, node E will send the confirmation APS message to the end-node A and forward the switching request APS message to its adjacent (downstream) node (say node F). When the confirmation APS message is received by node A, the cross-connection on node A is established. At this time the traffic is bridged to and selected from the...
Internet-Draft   GMPLS Extension for SMP   July 2019

protecting LSP at node A. After forwarding the switching request APS message, node E will wait for a confirmation APS message from node F, which triggers node E to set up the cross-connection for the protecting LSP being activated. If the protection resource is not available (due to failure or being used by higher priority connections), the switching will not be successful; the intermediate node may send a message to notify the end node, or may keep trying until the resource is available or the switching request is cancelled. If the resource is in use by a lower priority protecting LSP, the lower priority service will be removed and then the intermediate node will follow the procedure as described for the case when the protection resource is available for the higher priority protecting LSP.

The following subsections detail how LSPs using SMP can be signaled in an interoperable fashion using GMPLS RSVP-TE extensions (see RFC 3473 [RFC3473]). This includes;

(1) the ability to identify a "secondary protecting LSP" (hereby called the "secondary LSP") used to recover another primary working LSP (hereby called the "protected LSP")

(2) the ability to associate the secondary LSP with the protected LSP,

(3) the capability to include information about the resources used by the protected LSP while instantiating the secondary LSP,

(4) the capability to instantiate during the provisioning phase several secondary LSPs in an efficient manner, and

(5) the capability to support activation of a secondary LSP after failure occurrence via APS protocol in the data plane.

4.1. Identifiers

To simplify association operations, both LSPs (i.e., the protected and the secondary LSPs) belong to the same session. Thus, the SESSION object MUST be the same for both LSPs. The LSP ID, however, MUST be different to distinguish between the protected LSP carrying working traffic and the secondary LSP.

A new LSP Protection Type "Shared Mesh Protection" is introduced to the LSP Flags of PROTECTION object (see [RFC4872]) to set up the two LSPs. This LSP Protection Type value is applicable only to bidirectional LSPs as required in [G808.3].
4.2. Signaling Primary LSPs

The PROTECTION object (see [RFC4872]) is included in the Path message during signaling of the primary working LSPs, with the LSP Protection Type value set to "Shared Mesh Protection".

Primary working LSPs are signaled by setting in the PROTECTION object the S bit to 0, the P bit to 0, the N bit to 1 and in the ASSOCIATION object, the Association ID to the associated secondary protecting LSP_ID.

Note: N bit is set to indicate that the protection switching signaling is done via data plane.

4.3. Signaling Secondary LSPs

The PROTECTION object (see [RFC4872]) is included in the Path message during signaling of the secondary protecting LSPs, with the LSP Protection Type value set to "Shared Mesh Protection".

Secondary protecting LSPs are signaled by setting in the PROTECTION object the S bit and the P bit to 1, the N bit to 1 and in the ASSOCIATION object, the Association ID to the associated primary working LSP_ID, which MUST be known before signaling of the secondary LSP. Moreover, the Path message used to instantiate the secondary LSP SHOULD include at least one PRIMARY_PATH_ROUTE object (see [RFC4872]) that further allows for recovery resource sharing at each intermediate node along the secondary path.

With this setting, the resources for the secondary LSP SHOULD be pre-reserved, but not committed at the data plane level, meaning that the internals of the switch need not be established until explicit action is taken to activate this LSP. Activation of a secondary LSP and protection switching to the activated protecting LSP is done using APS protocol in the data plane.

After protection switching completes the protecting LSP SHOULD be signaled with the S bit set to 0 and O bit set to 1 in the PROTECTION object. At this point, the link and node resources must be allocated for this LSP that becomes a primary LSP (ready to carry normal traffic). The formerly working LSP MAY be signaled with the A bit set in the ADMIN_STATUS object (see [RFC3473]).

Support for extra traffic in SMP is for further study. Therefore, mechanisms to setup LSPs for extra traffic are also for further study.
The preemption priority of a protecting LSP that is used to resolve the competition for the same shared resource among multiple protecting LSPs, is indicated in the TBD1 field of the TBD2 object in the Path message of the protecting LSP. In SMP, the Setup and Holding priorities in the SESSION_ATTRIBUTE object can be used to configure or pre-configure a LSP, but is irrelevant to resolving the competition among multiple protecting LSPs, which experience failures on their working LSPs.

When an intermediate node on the protection LSP receives the Path message, the preemption priority value in the TBD1 field MUST be stored for that protection LSP. When resource competition among multiple protecting LSPs occurs, their priority values will be used to resolve the competition. Once the preemption priorities are configured, the preemption of the protecting LSPs is fully controlled by the APS.

When an APS request for a lower priority protecting LSP is preempted or cannot be confirmed due to existing higher priority APS request for another protection LSPs, an intermediate node MAY send PathErr and ResvErr with the error code/sub-code "Policy Control Failure/Hard Pre-empted" toward the source nodes of Path and Resv, respectively, to notify that the lower priority protecting LSP is preempted.

Upon receiving a PathErr or ResvErr message with the error code/sub-code "Policy Control Failure/Hard Pre-empted," the end node that has initiated the protection switching for a protecting LSP may cancel it (and try with another protecting LPS) or may keep trying until the resource is available.

In SMP, a preempted LSP SHOULD not be torn down. Once the working LSP and the protecting LSP are configured or pre-configured, the end node SHOULD keep refreshing both working and protecting LSPs regardless of failure or preempted situation.

[Editor’s note: See if it is ok to add the next sentence at the end of the previous paragraph.] The Path_State_Removed flag in the ERROR_SPEC object MUST not be set in PathErr and ResvErr messages generated due to preemption.

[Editor’s note: Check what should be the behavior to notify the end nodes of the lower priority protecting LSP that is no longer preempted and therefore it is available for SMP protection switching, if needed.]
4.4. SMP APS Configuration

SMP relies on APS protocol messages being exchanged between the nodes along the path to activate a SMP protecting LSP.

In order to allow exchange of APS protocol messages, an APS channel has to be configured between adjacent nodes along the path of the SMP protecting LSP. This should be done before any SMP protecting LSP has been setup by other means than GMPLS signaling which are therefore outside the scope of this document.

Depending on the APS protocol message format, the APS protocol may use different identifiers than GMPLS signaling to identify the SMP protecting LSP.

Since APS protocol is for further study in [G808.3], it can be assumed that APS message format and identifiers are technology-specific and/or vendor-specific. Therefore, additional requirements for APS configuration are outside the scope of this document.

5. Updates to PROTECTION Object

GMPLS extension requirements for SMP introduce several updates to the Protection Object (see [RFC4872]).

5.1. New Protection Type

A new LSP protection type "Shared Mesh Protection" is added in the protection object. This LSP Protection Type value is applicable to only bidirectional LSPs.

LSP (Protection Type) Flags:

0x11: Shared Mesh Protection

5.2. Other Updates

N bit and O bit in the Protection object as defined in [RFC4872] are also updated to include applicability to SMP.

Notification (N): 1 bit

When set to 1, this bit indicates that the control plane message exchange is only used for notification during protection switching. When set to 0 (default), it indicates that the control plane message exchanges are used for protection-switching purposes. The N bit is only applicable when the LSP Protection Type Flag is set to either 0x04 (1:N Protection with Extra-
Traffic), or 0x08 (1+1 Unidirectional Protection), or 0x10 (1+1 Bidirectional Protection). In SMP, N bit MUST be set to 1. The N bit MUST be set to 0 in any other case.

Operational (O): 1 bit

When set to 1, this bit indicates that the protecting LSP is carrying the normal traffic after protection switching. The O bit is only applicable when the P bit is set to 1, and the LSP Protection Type Flag is set to either 0x04 (1:N Protection with Extra-Traffic), or 0x08 (1+1 Unidirectional Protection), or 0x10 (1+1 Bidirectional Protection), or 0x11 (Shared Mesh Protection). The O bit MUST be set to 0 in any other case.

6. IANA Considerations

IANA actions required by this document will be described later.

7. Security Considerations

No further security considerations than [RFC4872].

8. Contributor

The following person contributed significantly to the content of this document and should be considered as a co-author.

Yuji Tochio
Fujitsu
Email: tochio@fujitsu.com

9. References

9.1. Normative References


9.2. Informative References


Authors’ Addresses

Jia He
Huawei Technologies
F3-1B, R&D Center, Huawei Industrial Base, Bantian, Longgang District Shenzhen China

Email: hejia@huawei.com

Italo Busi
Huawei Technologies

Email: italo.busi@huawei.com
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 January 9, 2020.

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.
1. Introduction

This document defines a YANG [RFC7950] data model for describing the relationship between a layer 3 network topology [RFC8346] and a Traffic Engineering (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 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 for L3 TE Topologies

The YANG modules ietf-l3-te-topology and ietf-l3-te-topology-state model configuration and operational state of layer 3 TE network topologies. These two modules augment ietf-l3-te-topology and ietf-l3-te-topology-state respectively, so a layer 3 TE network topology is a layer 3 network topology with additional TE capability enabled.

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 that 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:

```
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 parameters 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:

```
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:
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

The technology agnostic TE Topology model is defined in [I-D.ietf-teas-yang-te-topo], which is extended by this document to cover the Packet Switch Capable (PSC) technology [RFC3471] [RFC7074].

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], [RFC8570] 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
4. Complete Model Tree Structure

4.1. Layer 3 TE Topology Module

The model tree structure of the layer 3 TE topology module is as shown below:

```
module: ietf-l3-te-topology
  augment /nw:networks/nw:network/nw:network-types
    /l3t:l3-unicast-topology:
      +++-rw l3-te!
  augment /nw:networks/nw:network/l3t:l3-topology-attributes:
    +++-rw l3-te-topology-attributes
  augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
    +++-rw l3-te-node-attributes
      +++-rw node-ref? leafref
  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
  augment /nw:networks/nw:network/nt:link/l3t:l3-link-attributes:
    +++-rw l3-te-link-attributes
      +++-rw link-ref? leafref
```

4.2. Packet Switching TE Topology Module

This is an augmentation to base TE topology model.

```
module: ietf-te-topology-packet
  augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices:
      +++-ro performance-metrics-one-way
        +++-ro one-way-delay? uint32
        +++-ro one-way-delay-normality? te-types:performance-metrics-normality
```

Liu, et al.  Expires January 9, 2020
```yang
+--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?              _uint32
+--ro one-way-max-delay?                       uint32
   +--ro one-way-max-delay-normality?            uint32
+--ro one-way-delay-variation?                 uint32
   +--ro one-way-delay-variation-normality?     uint32
+--ro one-way-packet-loss?                     decimal64
   +--ro one-way-packet-loss-normality?         decimal64
+--ro performance-metrics-two-way
   +--ro two-way-delay?                       uint32
   +--ro two-way-delay-normality?              uint32
   +--ro two-way-min-delay?                   uint32
   +--ro two-way-min-delay-normality?          uint32
   +--ro two-way-max-delay?                   uint32
   +--ro two-way-max-delay-normality?          uint32
   +--ro two-way-delay-variation?              uint32
   +--ro two-way-delay-variation-normality?    uint32
   +--ro two-way-packet-loss?                 decimal64
   +--ro two-way-packet-loss-normality?       decimal64
+--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?
```
Internet-Draft             YANG L3 TE Topology                 July 2019

| 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: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?
        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
  +--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
++--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:information-source-entry/tet:connectivity-matrices:
++--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 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
++-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-delay-variation?       uint32
  +--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 two-way-min-delay?      uint32
++-ro two-way-max-delay?      uint32
++-ro two-way-delay-variation?       uint32
++-ro two-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-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
++-ro two-way-delay?          uint32
++-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 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:information-source-entry/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?
        |    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?
    ++--ro one-way-min-delay-normality?
        |    te-types:performance-metrics-normality
    ++--ro one-way-max-delay?
    ++--ro one-way-max-delay-normality?
        |    te-types:performance-metrics-normality
    ++--ro one-way-delay-variation?
    ++--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

---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
Internet-Draft             YANG L3 TE Topology                 July 2019

| +--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-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?
|       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?

++-ro performance-metrics-one-way
  +--ro one-way-delay-normality?              uint32
  +--ro one-way-residual-bandwidth-normality?
    |  te-types:performance-metrics-normality
  +--ro one-way-available-bandwidth-normality?
    |  te-types:performance-metrics-normality
  +--ro one-way-utilized-bandwidth-normality?
    |  te-types:performance-metrics-normality
  +--ro one-way-min-delay?                    uint32
  +--ro one-way-max-delay?
  +--ro one-way-delay-variation?
  +--ro one-way-packet-loss?
  +--ro two-way-delay-normality?
  +--ro two-way-residual-bandwidth-normality?
  +--ro two-way-available-bandwidth-normality?
  +--ro two-way-utilized-bandwidth-normality?
  +--ro two-way-min-delay?
  +--ro two-way-max-delay?
  +--ro two-way-delay-variation?
  +--ro two-way-packet-loss?
  +--ro threshold-accelerated-advertisement
    +--ro one-way-delay-normality?
    +--ro one-way-residual-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro one-way-available-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro one-way-utilized-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro one-way-min-delay?                    uint32
    +--ro one-way-max-delay?
    +--ro one-way-delay-variation?
    +--ro one-way-packet-loss?
    +--ro two-way-delay-normality?
    +--ro two-way-residual-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro two-way-available-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro two-way-utilized-bandwidth-normality?
      |  te-types:performance-metrics-normality
    +--ro two-way-min-delay?
---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

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
    te-types:performance-metrics-normality
  +-ro one-way-max-delay?
    te-types:performance-metrics-normality
  +-ro one-way-max-delay-normality?
    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
    +-ro two-way-delay-normality?
      te-types:performance-metrics-normality
    +-ro two-way-min-delay?
      te-types:performance-metrics-normality
    +-ro two-way-min-delay-normality?
      te-types:performance-metrics-normality
    +-ro two-way-max-delay?
      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
  +-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
    +--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
|    |    +--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 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-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
      +--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
   +--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?          uint32
  |  ++--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?          uint32
  |  ++--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?          uint32
  |  ++--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
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@2019-06-28.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
Internet-Draft             YANG L3 TE Topology                 July 2019

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 2019-06-28 {

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/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";
    }
    description
    "Augment link configuration";
    uses l3-te-link-attributes;
}

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.";
    }
    description
    "Containing TE topology references";
    uses nw:network-ref;
} // 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.";
  }
  description
  "Containing TE node references";
  uses nw:node-ref;
} // l3-te
} // 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.";
      }
    description
    "Containing TE termination point references";
    uses nt:tp-ref;
  } // l3-te
} // l3-te-tp-attributes

grouping l3-te-link-attributes {
  description
  "L3 TE link scope attributes";
  container l3-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;
  } // l3-te
} // l3-te-link-attributes
5.2. Packet Switching TE Topology Module

This module references [RFC7471], [RFC7823], [RFC8294], [RFC8345], [RFC8346], [RFC8570], [I-D.ietf-teas-yang-te-types], and [I-D.ietf-teas-yang-te-topo].

<CODE BEGINS> file "ietf-te-topology-packet@2019-06-28.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
  }

<CODE ENDS>
"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"
Internet-Draft             YANG L3 TE Topology                 July 2019

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 2019-06-28 {
  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
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
    RFC7823: Performance-Based Path Selection for Explicitly
    Routed Label Switched Paths (LSPs) Using TE Metric
    Extensions";
}

/*
 * 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;
    }
    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:information-source-entry/tet:connectivity-matrices" {

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;
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
  + "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;
      }
      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-pscl’]"
    
description
  "Valid only for PSC";
}

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

/* Augmentations to interface-switching-capability */
augment "/nw:networks/tet:te/tet:templates/"
  + "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
      te-packet-types:performance-metrics-attributes-packet {
        if-feature te-performance-metric;
      }
      te-packet-types:performance-metrics-throttle-container-packet {
        if-feature te-performance-metric;
      }
    }
  }
uses packet-switch-capable-container;
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
   + "tet:te-link-attributes/"
   + "tet:interface-switching-capability" {
    when "tet:switching-capability = 'te-types:switching-psc1' " {
     description
     "Valid only for PSC";
    }

description
"Parameters for PSC TE topology.";
uses packet-switch-capable-container;
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
   + "tet:information-source-entry/"
   + "tet:interface-switching-capability" {
    when "tet:switching-capability = 'te-types:switching-psc1' " {
     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]:

--------------------------------------------------------------------
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------
This document registers the following YANG modules in the YANG Module Names registry [RFC6020):

name:         ietf-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

name:         ietf-te-topology-packet-state
prefix:       tet-pkt-s
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 [RFC8446].

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.

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/networks/nw:network/nw:network-types/l3t:l3-unicast-topology/
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.

/nw/networks/nw: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.

/nw/networks/nw:network/nw: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.

/nw/networks/nw:network/nw:node/nt:termination-point//l3t:l3-
termination-point-attributes/l3-te-tp-attributes
  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.

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.

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:

Unauthorized access to this subtree can disclose the layer 3 TE topology type.

Unauthorized access to this subtree can disclose the topology-wide configurations, including the reference to a TE topology from a layer 3 network topology.

Unauthorized access to this subtree can disclose the operational state information of layer 3 TE nodes.

Unauthorized access to this subtree can disclose the operational state information of layer 3 TE link termination points.
Unauthorized access to this subtree can disclose the operational state information of layer 3 TE links.

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@2019-06-28.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 2019-06-28 {
  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/nw-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" {

A.2. Packet Switching TE Topology State Module

<CODE BEGINS> file "ietf-te-topology-packet-state@2019-06-28.yang"
module ietf-te-topology-packet-state {
  yang-version 1.1;
  namespace

  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 2019-06-28 {
  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/"
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;
    }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/
+ "tet-s:tunnel-termination-point/"
+ "tet-s:local-link-connectivities/"
+ "tet-s:local-link-connectivity" {
    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:site/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-pscl']"
    {
        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:site/"
    + "tet-s:te-link-attributes"
    when "tet-s:interface-switching-capability"
    + "[tet-s: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 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:site/"
    + "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-psc1' " {
      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-psc1' " {
      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-psc1' " {
      description
        "Valid only for PSC";
    }
  description
    "Parameters for PSC TE topology."
  uses tet-pkt:packet-switch-capable-container;
}
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.

```
{  
    "ietf-network:networks": {  
        "network": [  
            {  
                "network-id":"example-topo-te",  
                "network-types": {  
                    "ietf-te-topology:te-topology": {  
                    }  
                },  
                "ietf-te-topology:te-topology-identifier": {  
                    "L1": "example-topo-te",  
                    "L2": "example-topo-te",  
                    "L3": "example-topo-te"  
                }  
            }  
        ]  
    }  
}
```

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.
"provider-id": 200,
"client-id": 300,
"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":
"ietf-te-types:switching-psc1",
"encoding":
"ietf-te-types:lsp-encoding-ethernet"
}
]
}
},
{
"tp-id": "1-3-1",
"ietf-te-topology:te-tp-id": 10301,
"ietf-te-topology:te": {
"interface-switching-capability": [
{
"switching-capability":
"ietf-te-types:switching-psc1",
"encoding":
"ietf-te-types:lsp-encoding-ethernet"
}
]
}
]
},
"node-id": "D2",
"ietf-te-topology:te-node-id": "2.0.2.1",
"ietf-te-topology:te": {}
"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": "ietf-te-types:switching-pscl",
          "encoding": "ietf-te-types: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": "ietf-te-types:switching-pscl",
          "encoding": "ietf-te-types:lsp-encoding-ethernet"
        }
      ]
    }
  }
],
"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-te-topology:te": { 
  "te-link-attributes": { 
    "interface-switching-capability": [ 
      { 
        "switching-capability": 
          "ietf-te-types:switching-pscl",
        "encoding": "ietf-te-types: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": 
          "ietf-te-types:switching-pscl",
        "encoding": "ietf-te-types:lsp-encoding-ethernet"
      }
    ],
    "max-link-bandwidth": { 
      "te-bandwidth": { 
        "generic": "0x1p+18"
      }
    },
    "te-default-metric": 100
  }
}
"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": "ietf-te-types:switching-pscl",
 "encoding":"ietf-te-types: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": {
 "te-link-attributes": {
 "interface-switching-capability": [ 

"switching-capability":
   "ietf-te-types:switching-pscl",
   "encoding":"ietf-te-types:lsp-encoding-ethernet"
],
"max-link-bandwidth": {
   "te-bandwidth": {
      "generic":"0x1p+18"
   },
   "te-default-metric":100
},

"network-id":"example-topo-l3-te",
"network-types": {
   "ietf-l3-unicast-topology:l3-unicast-topology": {
      "ietf-l3-te-topology:l3-te": {
      
   
   },
   "ietf-l3-unicast-topology:l3-topology-attributes": {
      "ietf-l3-te-topology:l3-te-topology-attributes": {
         "network-ref":"example-topo-te"
      
      },
      "node": [
      
      "node-id":"D1",
      "ietf-l3-unicast-topology:l3-node-attributes": {
         "router-id": [
            "203.0.113.1"
         ],
         "prefix": [
            "prefix":"203.0.113.1/32"
         ],
      "ietf-l3-te-topology:l3-te-node-attributes": {
         "node-ref":"D1",
         "network-ref":"example-topo-te"
      
      },
      "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": {
      "network-ref": "example-topo-te",
      "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": {
      "network-ref": "example-topo-te",
      "tp-ref": "1-3-1"
    }
  }
}
{
  "node-id": "D2",
  "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",
    "network-ref": "example-topo-te"
  }
}
"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-13-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id":211,
"ietf-13-te-topology:l3-te-tp-attributes": {
"tp-ref":"2-1-1",
"network-ref":"example-topo-te"
}
},
{
"tp-id":"2-3-1",
"ietf-13-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id":231
}
}
},
{
"node-id":"D3",
"ietf-13-unicast-topology:l3-node-attributes": {
"router-id": [ 
"203.0.113.3"
],
"prefix": [ 
{"prefix":"203.0.113.3/32"}
]
},
"ietf-13-te-topology:l3-te-node-attributes": {
"node-ref":"D3",
"network-ref":"example-topo-te"
}
}
,"ietf-network-topology:termination-point": [{
"tp-id":"3-0-1",
"ietf-13-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id":301
}
},
{
"tp-id":"3-1-1",
"ietf-13-unicast-topology:l3-termination-point-attributes": {
"unnumbered-id":311,
"ietf-l3-te-topology:l3-te-tp-attributes": {
  "tp-ref":"3-1-1",
  "network-ref":"example-topo-te"
}
},
{"tp-id":"3-2-1",
 "ietf-l3-unicast-topology:l3-termination-point-attributes": {
  "unnumbered-id":321
}
}]}
"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",
    "network-ref":"example-topo-te"
  }
 }
 },
 {
 "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",
    "network-ref":"example-topo-te"
  }
 }
]
"network-ref": "example-topo-te"
]
}
},
{
"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-l3-te-topology:l3-te-link-attributes": {
"link-ref": "D1,1-3-1,D3,3-1-1",
"network-ref": "example-topo-te"
}
}
},
{
"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-l3-te-topology:l3-te-link-attributes": {
"link-ref": "D3,3-1-1,D1,1-3-1",
"network-ref": "example-topo-te"
}
}
},
{
"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-l3-te-topology:l3-te-link-attributes": {
"link-ref": "D2,2-3-1,D3,3-2-1",
"network-ref": "example-topo-te"
}
}
"dest-tp":"3-2-1"
},
"ietf-l3-unicast-topology:l3-link-attributes": {
  "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
Futurewei
EMail: igor.bryskin@futurewei.com

Vishnu Pavan Beeram
Juniper Networks
EMail: vbeeram@juniper.net
A YANG Data Model for Resource Reservation Protocol (RSVP)
draft-ietf-teas-yang-rsvp-11

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 RSVP extensions to Traffic-Engineering
(RSVP-TE). The model covers the configuration, operational state,
remote procedure 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 January 5, 2020.

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
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 "rt: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.iotf-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 and are supported by most vendors claiming the 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 ability to better tune the 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.

![Diagram of module relationships]

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 the configuration for optional features that are not must for basic RSVP protocol 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. Model Notifications

Notifications data modeling is key in any defined data model.

[I-D.ietf-netconf-subscribed-notifications] and [I-D.ietf-netconf-yang-push] define a subscription and push mechanism for YANG datastores. This mechanism currently allows the user to:

- Subscribe 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.

2.4. 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.

The derived state data is contained in "read-only" nodes directly under the intended object as shown in Figure 2.
module: ietf-rsvp
   +--rw rsvp!
      +--rw globals
         .
         .
      +--rw interfaces
         .
         .
      +--rw neighbors
         .
         .
      +--rw sessions
         .
         .
      rpcs:
         +--x clear-session
         +--x clear-neighbor

Figure 2: RSVP high-level tree model view

Configuration and state data are grouped to those applicable on per node (global), per interface, per neighbor, or per session.

Global Data:

The global data cover the configuration and state that is applicable the RSVP protocol behavior.

Interface Data:

The interface data configuration and state model relevant attributes applicable to one or all RSVP interfaces. Any data or state at the "interfaces" container level is equally applicable to all interfaces - unless overridden by explicit configuration or state under a specific interface.

Neighbor Data:

The neighbor data cover configuration and state relevant to RSVP neighbors. Neighbors can be dynamically discovered using RSVP signaling or explicitly configured.
Session Data:

The sessions data branch covers configuration and state relevant to RSVP sessions. This is usually derived state that is result of signaling. This model defines attributes related to IP RSVP sessions as defined in [RFC2205].

2.4.1. Tree Diagram

Figure 3 shows the YANG tree representation for configuration and state data that is augmenting the RSVP basic module:

```yang
module ietf-rsvp
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol:
      +-rw rsvp
        +-rw globals
        |  +-rw sessions
        |     +-ro session-ip
        |     |  [destination protocol-id destination-port]
        |     |     +-ro destination-port  inet:port-number
        |     |     +-ro protocol-id       uint8
        |     |     +-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
        |     +-ro statistics
        |     |  +-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
```

Beeram, et al. Expires January 5, 2020
++-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-length? 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
++-ro statistics
| ++-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-length?  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
  |  |  |  +++-ro statistics
  |  |  |  |  +++-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-length?  yang:counter64
++--rw neighbors
++--rw neighbor* [address]
++--rw address  inet:ip-address
++--rw epoch?  uint32
++--rw expiry-time?  uint32
++--rw graceful-restart
++--rw enabled?  boolean
++--rw local-restart-time?  uint32
++--rw local-recovery-time?  uint32
++--rw neighbor-restart-time?  uint32
++--rw neighbor-recovery-time?  uint32
++--rw helper-mode
++--rw enabled?  boolean
++--rw max-helper-restart-time?  uint32
++--rw max-helper-recovery-time?  uint32
++--rw neighbor-restart-time-remaining?  uint32
++--rw neighbor-recovery-time-remaining?  uint32
++--rw hello-status?  enumeration
++--rw interface?  if:interface-ref
++--rw neighbor-state?  enumeration
++--rw refresh-reduction-capable?  boolean
2.4.2. YANG Module

The ietf-rsvp module imports from the following modules:

- ietf-interfaces defined in [RFC8343]
- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-routing defined in [RFC8349]
- ietf-key-chain defined in [RFC8177]

<CODE BEGINS> file "ietf-rsvp@2019-07-04.yang"
module ietf-rsvp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp";

  /* Replace with IANA when assigned */
  prefix "rsvp";
}

Figure 3: RSVP model tree diagram
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>"
  "Editor:  Vishnu Pavan Beeram  
    <mailto:vbeeram@juniper.net>"
  "Editor:  Tarek Saad  
    <mailto:tsaad@juniper.net>"
  "Editor:  Rakesh Gandhi  
    <mailto:rgandhi@cisco.com>"
  "Editor:  Xufeng Liu  
    <mailto:xufeng.liu.ietf@gmail.com>"
  "Editor:  Igor Bryskin"
This module contains the RSVP YANG data model. The model fully conforms to the Network Management Datastore Architecture (NMDA).

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

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

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

// 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 "2019-07-04" {
    description
        "A YANG Data Model for Resource Reservation Protocol"
    reference
        "RFCXXXX: A YANG Data Model for Resource Reservation Protocol (RSVP)"
}

identity rsvp {
    base "rt:routing-protocol"
    description "RSVP protocol"
}

identity rsvp-session-type {
    description "Base RSVP session type"
}

identity rsvp-session-ip {
    base rsvp-session-type
    description "RSVP IP 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 Graceful-Restart";
    leaf enabled {
        type boolean;
        description
            "'true' if RSVP Graceful Restart is enabled.
             'false' if RSVP Graceful Restart is disabled.";
    }
}

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

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";
    container 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 destination-port {
  type inet:port-number;
  description "RSVP destination port";
  reference "RFC2205";
}
leaf protocol-id {
  type uint8;
  description "The IP protocol ID.";
  reference "RFC2205, section 3.2";
}
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";
    reference "RFC2205";
  }
  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";
      reference "RFC2205";
    }
    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";
  }
}
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.";
  }
}

} // helper-mode
} // graceful-restart
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 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;
  description
  "Integrity Challenge sent count";
}

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;
    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-length {
type yang:counter64;
description
"The total number of packets received with an invalid packet length."
}
}

grouping statistics-state {
description "RSVP statistic attributes.";
container statistics {
config false;
description
"statistics state container";
uses protocol-state;
uses packets-state;
uses errors-state;
}
}

grouping neighbor-derived-state {
description
"Derived state at neighbor level."
}

grouping global-attributes {
description
"Top level grouping for RSVP global properties";
container sessions {
description
"RSVP sessions container";
list session-ip {
key "destination protocol-id destination-port";
config false;
description
"List of RSVP sessions";
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.";
    leaf interface {
      type if:interface-ref;
      description
        "RSVP interface.";
      }
    uses intf-attributes;
  }
}
container neighbors {
  description "RSVP neighbors container";
  list neighbor {
    key "address";
    description "List of RSVP neighbors";
    uses neighbor-attributes;
  }
}

grouping session-ref {
  description "Session reference information";
  leaf destination {
    type leafref {
    } mandatory true;
  description "RSVP session";
  }
  leaf protocol-id {
    type uint8;
    mandatory true;
    description "The RSVP session protocol ID";
  }
  leaf destination-port {
    type inet:ip-address;
    mandatory true;
    description "The RSVP session destination port";
  }
}

rpc clear-session {
  description "Cleares RSVP sessions RPC";
  input {
    leaf routing-protocol-instance-name {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rt:name";
      } mandatory "true";
      description "Name of the RSVP protocol instance whose session is being cleared.
      If the corresponding RSVP instance doesn’t exist,
then the operation will fail with an error-tag of 'data-missing' and an error-app-tag of 'routing-protocol-instance-not-found'.
}
}
}
}
}
}
}
}
}

rpc clear-neighbor {
  description
    "RPC to clear the RSVP Hello session to a neighbor";
  input {
    leaf routing-protocol-instance-name {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/" + "rt:control-plane-protocol/rt:name";
      } mandatory "true";
      description
        "Name of the RSVP protocol instance whose session is being cleared.

        If the corresponding RSVP instance doesn’t exist, then the operation will fail with an error-tag of 'data-missing' and an error-app-tag of"
2.5. 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.

2.5.1. Tree Diagram

Figure 4 shows the YANG tree representation for configuration and state data that is augmenting the RSVP extended module:

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:packets:
  +-ro discontinuity-time? yang:date-and-time
  +-ro out-dropped? yang:counter64
  +-ro in-dropped? yang:counter64
  +-ro out-errors? yang:counter64
  +-ro in-errors? yang:counter64
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
    /rsvp:statistics/rsvp:messages:
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
    /rsvp:statistics/rsvp:errors:
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces:
    +-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:rsvp/rsvp:interfaces
    /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:rsvp/rsvp:interfaces
    /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:rsvp/rsvp:interfaces
    /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:rsvp/rsvp:interfaces
    /rsvp:interface:
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
        /rsvp:interface/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:rsvp/rsvp:interfaces
        /rsvp:interface/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:rsvp/rsvp:interfaces
        /rsvp:interface/rsvp:authentication:
            +++rw lifetime?      uint32
            +++rw window-size?   uint32
            +++rw challenge?     empty
            +++rw retransmits?   uint32
            +++rw key-chain?     key-chain:key-chain-ref

Figure 4: RSVP extended model tree diagram

2.5.2. YANG Module

The ietf-rsvp-extended module imports from the following modules:

- ietf-rsvp defined in this document
- ietf-routing defined in [RFC8349]
- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-key-chain defined in [RFC8177]

Figure 5 shows the RSVP extended YANG module:

```<CODE BEGINS> file "ietf-rsvp-extended@2019-07-04.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";
}

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>
    Editor:   Vishnu Pavan Beeram
               <mailto:vbeeram@juniper.net>
    Editor:   Tarek Saad
               <mailto:tsaad@juniper.net>
    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
This module contains the Extended RSVP YANG data model. The model fully conforms to the Network Management Datastore Architecture (NMDA).

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

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

revision "2019-07-04" {
    description
        "A YANG Data Model for Extended Resource Reservation Protocol";
    reference
        "RFCXXXX: A YANG Data Model for 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 "0..16777215";
        }
        description "Number of retransmits for each security association";
        reference "RFC 2747: RSVP Cryptographic Authentication";
    }
}

range "1..10000";
}
description
  "Number of retransmits when messages are dropped.";
reference
  "RFC 2747: RSVP Cryptographic Authentication";
}
leaf key-chain {
  type key-chain:key-chain-ref;
  description
    "Key chain name to authenticate RSVP signaling messages.";
  reference
    "RFC 2747: RSVP Cryptographic Authentication";
}
}
grouping hellos-extended-config {
  description
    "Configuration parameters relating to RSVP hellos";
  leaf interface-based {
    type empty;
    description
      "Enable interface-based Hello adjacency if present.";
  }
  leaf hello-interval {
    type uint32;
    units milliseconds;
    description
      "Configure interval between successive Hello messages in milliseconds.";
    reference
  }
  leaf hello-misses {
    type uint32 {
      range "1..10";
    }
    description
      "Configure max number of consecutive missed Hello messages.";
grouping signaling-parameters-extended-config {
    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
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 {
    type yang:counter64;
    description
    "In packet drop count";
}

leaf out-errors {
    type yang:counter64;
    description
    "Out packet errors count";
}

leaf in-errors {
    type yang:counter64;
    description
    "In packet rx errors 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/rsvp:globals/" +
"rsvp:statistics/rsvp:packets" {
description
"RSVP packet stats extensions";
uses packets-extended-state;
}
augment "/rt:routing/rt:control-plane-protocols/" +
"rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" +
"rsvp:statistics/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/rsvp:globals/" +
"rsvp:statistics/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/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/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/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/
    description
    "RSVP refresh-reduction interface configuration extensions";
uses refresh-reduction-extended-config;
}

/* RSVP hellos extension */
augment "/rt:routing/rt:control-plane-protocols/"
Internet-Draft            RSVP YANG Data Model                 July 2019

/* 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;
   }

} <CODE ENDS>

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 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 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.

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.

/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:

The presence of this container enables the RSVP protocol functionality on a device. It also controls the configuration settings on data nodes pertaining to RSVP sessions, interfaces and neighbors. All of which are considered sensitive and if access to either of these is compromised, it can result in temporary network outages or be employed to mount DoS attacks.

For RSVP authentication, the configuration supported is via the specification of key-chains [RFC8177] or the direct specification of key and authentication algorithm, and hence security considerations of [RFC8177] are inherited. This includes the considerations with respect to the local storage and handling of authentication keys.

Some of the RPC operations defined in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. The RSVP YANG module support the "clear-session" and "clear-neighbor" RPCs. If access to either of these is compromised, they can result in temporary network outages be employed to mount DoS attacks.

The security considerations spelled out in the YANG 1.1 specification [RFC7950] apply for this document as well.
5. Acknowledgement

The authors would like to thank Lou Berger for reviewing and providing valuable feedback on this document.

6. Contributors

Himanshu Shah
Ciena
Email: hshah@ciena.com

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-netconf-subscribed-notifications]

[I-D.ietf-netconf-yang-push]


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
Jabil
Email: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei Technologies
Email: Igor.Bryskin@huawei.com
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 are 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
basis of implementation for other interfaces, such as CLI and programmatic APIs.

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] 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.

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
---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:control-plane-protocol/rsvp:rsvp/rsvp:neighbors:
  /rt:neighbors/rsvp-neighbors/rsvp-neighbors:
  ---ro associated-rsvp-session?            leafref
  ---ro lsp-signaled-name?                  string
  ---ro lsp-attribute*                     identityref
  ---ro lsp-signaled-name?                  string
  ---ro session-attribute*                  identityref
  ---ro session-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 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
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 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 direction?  te-label-direction
            ++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
<table>
<thead>
<tr>
<th></th>
<th>++--ro flags*     path-attribute-flags</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>++--:(unnumbered-link-hop)</td>
</tr>
<tr>
<td></td>
<td>++--ro unnumbered-link-hop</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>++--:(label)</td>
</tr>
<tr>
<td></td>
<td>++--ro label-hop</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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:
|            | ++--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 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

augment /te:te/tunnels/te:tunnel/te:p2p-secondary-paths
/te:p2p-secondary-path/te:lsp-provisioning-error-infos
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.

module: ietf-rsvp-te-mpls
augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp:
    +--rw fast-reroute-local-revertive
    +--rw rsvp-frr-local-revert-delay?   uint32
augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp/interfaces:
augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp/interfaces/rsvp:interface:
    /rsvp:sessions/rsvp:session/rsvp:state:
augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/rsvp:rsvp:neighbors:
augment /te:te/te:tunnels/te:tunnel:
    +--rw session-attribute*   identityref
augment /te:te/te:lsps-state/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:tunnels/te:tunnel/te:p2p-primary-paths/te:p2p-primary-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:tunnels/te:tunnel/te:p2p-secondary-paths
tep2p-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 percent-value? uint32
  ++--rw (bc-model-type)?
    |  ++--:(bc-model-rdm)
    |    ++--rw bc-model-rdm
    |    |  ++--rw bandwidth-mpls-constraints
    |    |    ++--rw maximum-reservable?
    |    |    |  ++--te-packet-types:bandwidth-kbps
    |    |    ++--rw bc-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]
    |    |    ++--rw 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].

<CODE BEGINS> file "ietf-rsvp-te@2019-07-06.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";
}

import ietf-inet-types {
  prefix inet;
  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>

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>
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 "2019-07-06" {
  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";
    ordered-by user;
    description "List of RSVP Path record-route objects";
    uses te-types:record-route-state;
}
}
}
}
}
}
}
}
}
}

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;
        }
    }
}
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
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 */
      "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" {
  description
  "RSVP-TE generic data augmentation pertaining to session";
  list session-te {
    key "tunnel-endpoint tunnel-id extended-tunnel-id";
    config false;
    description
    "List of RSVP sessions";
    leaf tunnel-endpoint {
      type inet:ip-address;
      description "XX";
    }
    leaf tunnel-id {
      type uint16;
      description "XX";
    }
    leaf extended-tunnel-id {
      type inet:ip-address;
      description "XX";
    }

    uses rsvp:session-attributes-state;
  }
}

augment "/rt:routing/rt:control-plane-protocols/"
+ "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/
  rsvp:sessions/session-te/psbs/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";
}

  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";
  }
}
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" {
    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/rsvp:globals/
        + "rsvp:sessions/session-te/tunnel-id";
    }
    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" {
    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
  "When the signaling protocol is RSVP-TE ";
}
description
"RSVP-TE generic data augmentation pertaining to specific TE LSP";
uses rsvp-te-lsp-properties;
}

  description
  "Augmentation for RSVP-TE per LSP error reason";
  uses rsvp-te-lsp-error-info;
}

    description
    "When the signaling protocol is RSVP-TE ";
  }
description
  "Augmentation for RSVP-TE per path error reason";
  uses rsvp-te-lsp-error-info;
}

    description
    "When the signaling protocol is RSVP-TE ";
RSVP-TE generic data augmentation pertaining to specific TE LSP;
uses rsvp-te-lsp-properties;
}
augment "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
"/te:p2p-secondary-path" + 
"/te:lsp-provisioning-error-infos" + 
"/te:lsp-provisioning-error-info" 

description
"Augmentation for RSVP-TE per path error reason";
uses rsvp-te-lsp-error-info;
}
augment "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
"/te:p2p-secondary-path/te:lsps/te:lsp" + 
"/te:lsp-provisioning-error-infos" + 
"/te:lsp-provisioning-error-info" 
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
"Augmentation for RSVP-TE per LSP error reason";
uses rsvp-te-lsp-error-info;
}

/* TE interface augmentation */
augment "/te:te-dev:interfaces/te-dev:interface" 

description
"RSVP-TE generic data augmentation pertaining to specific TE interface";
}

<CODE ENDS>

Figure 4: RSVP TE generic YANG module

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]
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].

<CODE BEGINS> file "ietf-rsvp-te-mpls@2019-07-06.yang"
module ietf-rsvp-te-mpls {
    yang-version 1.1;
    prefix "rsvp-te-mpls";

    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-te-packet-types {
        prefix "te-packet-types";
        reference "draft-ietf-teas-yang-te-types: A YANG Data Model for
Common Traffic Engineering 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-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
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
/* RSVP-TE MPLS LSPs groupings */
grouping lsp-attributes-flags-mpls-config {
    description "Configuration parameters relating to RSVP-TE MPLS LSP 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 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;
}
grouping lsp-properties-mpls {
  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 preeemptions state grouping";
  container interface-softpreemption-state {
    description "The RSVP-TE interface preeemptions 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:lsps-state/te:lsp/"+
        "te:source";
      }
    }
    description "Tunnel sender address extracted from SENDER_TEMPLATE object";
  }
}
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";
        reference "RFC3209";
    }
}


```yang

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.";
}
}
}

/* 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";
                }
            }
        }
    }
    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";
}
}

/* 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;
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" { 
  description
  "Augmentation 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:lsps-state/te:lsp" { 
  when "/te:te:lsps-state/te:lsp" + 
  "/te:path-setup-protocol = 'te-types:path-setup-rsvp'" { 
  description
  "Augmentations for RSVP-TE MPLS TE tunnel properties";
  uses tunnel-properties-mpls-config;
}
"When the signaling protocol is RSVP-TE ";
)
description
  "RSP-TE MPLS LSP state properties";
  uses lsp-properties-mpls;
  uses lsp-backup-info;
}

  "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;
}

  "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;
}
}
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 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 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

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. 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


Internet-Draft      RSVP-TE Protocol YANG Data Model           July 2019

[ RFC4859 ]  Farrel, A., "Codepoint Registry for the Flags Field in the
Resource Reservation Protocol-Traffic Engineering (RSVP-
TE) Session Attribute Object", RFC 4859,
DOI 10.17487/RFC4859, April 2007,

and G. Ash, "Crankback Signaling Extensions for MPLS and
GMPLS RSVP-TE", RFC 4920, DOI 10.17487/RFC4920, July 2007,

Ayyangarps, "Encoding of Attributes for MPLS LSP
Establishment Using Resource Reservation Protocol Traffic
Engineering (RSVP-TE)", RFC 5420, DOI 10.17487/RFC5420,

Wright, "Label Switched Path (LSP) Attribute in the
Explicit Route Object (ERO)", RFC 7570,
DOI 10.17487/RFC7570, July 2015,

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

YANG Data Model for SR and SR TE Topologies
draft-ietf-teas-yang-sr-te-topo-05

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 January 9, 2020.

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 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 ........................................... 6
   2.7. Link Attributes ........................................... 6
3.  Model Structure ................................................ 7
4.  YANG Module .................................................... 9
5.  IANA Considerations .......................................... 17
6.  Security Considerations ....................................... 17
7.  References ..................................................... 19
   7.1. Normative References ..................................... 19
   7.2. Informative References ................................... 20
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.

```
+------------------------+
<p>|       SR Topology     |</p>
<table>
<thead>
<tr>
<th>ietf-sr-topology</th>
</tr>
</thead>
</table>
```

```
+------------------------+
<table>
<thead>
<tr>
<th>ietf-l3-unicast-topology</th>
</tr>
</thead>
</table>
```

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)
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te

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:
The SR attributes that are related to an IGP-Prefix segment are modeled by augmenting the list entry "prefix" in the L3 topology model:

```
  augment /nw:networks/nw:network/nw:node/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:

```
  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 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 usage of the leaf "advertise-protection" is described in [I-D.ietf-spring-sr-yang].

Both IGP and BGP can be supported by the model, the leaf "information-source" is used to indicate where the information is from.

The bundling capability of the Adjacency Segment is achieved by re-using the existing modeling construct (i.e. "bundle-stack-level") under /nw:networks/nw:network/nt:link/tet:te [I-D.ietf-teas-yang-te-topo]

3. Model Structure

The model tree structure of the Segment Routing (SR) topology module is as shown below:

```yin
module: ietf-sr-topology
  augment /nw:networks/nw:network/nw:network-types
    /l3t:13-unicast-topology:
      +++-rw sr-mpls!
    augment /nw:networks/nw:network/l3t:13-topology-attributes:
      +++-rw sr
        +++-rw srgb* [lower-bound upper-bound]
          +++-rw lower-bound uint32
          +++-rw upper-bound uint32
    augment /nw:networks/nw:network/nw:node/l3t:13-node-attributes:
      +++-rw sr
        +++-rw srgb* [lower-bound upper-bound]
          +++-rw lower-bound uint32
```
Internet-Draft             YANG SR TE Topology                 July 2019

    | ---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-instance?    string
    ---ro information-source-state
    | ---ro credibility-preference?   uint16
augment /nw:networks/nw:network/nw:node/l3t:13-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:13-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@2019-06-28.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

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 2019-06-28 {
  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;
}
augment "/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;
}
augment "/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;
}
augment "/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;
}
augment "/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 {
      refine srgb {
        must "lower-bound <= upper-bound" {

error-message
  "lower-bound must not be greater than upper-bound.";
}
}
} // 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-ls" {
      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

grouping sr-node-attributes {
 description "SR node scope attributes.";
 container sr {
 description
 "Containing SR attributes.";
 uses sr-cmn:srgb {
 refine srgb {
 must "lower-bound <= upper-bound" {
 error-message
 "lower-bound must not be greater than upper-bound.";
 }
 }
 uses sr-cmn:srlb {
 refine srlb {
 must "lower-bound <= upper-bound" {
 error-message
 "lower-bound must not be greater than upper-bound.";
 }
 }
 uses sr-cmn:node-capabilities;
 leaf msd {
 if-feature "msd";
type uint8;
description
"..."
"Node MSD is the lowest MSD supported by the node."
{
// Operational state data
uses information-source-attributes;
} // sr
} // sr-node-attributes

grouping sr-node-prefix-attributes {
  description "Containing SR attributes for a prefix.";
  container sr {
    presence "Presence indicates SR is enabled.";
    description
    "Containing SR attributes for a prefix.";
    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-readvertisment {
      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;
  }
} // sr
} // sr-node-attributes
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."
}
}
default "single";
description
"If set, the Adj-SID refers to an adjacency being protected."
}

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]:

--------------------------------------------------------------------
name:         ietf-sr-topology
prefix:       srt
reference:    RFC XXXX
--------------------------------------------------------------------

--------------------------------------------------------------------
name:         ietf-sr-topology-state
prefix:       srt-s
reference:    RFC XXXX
--------------------------------------------------------------------

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 [RFC8446].
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.

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.

/nw:networks/nw:node/l3t:l3-node-attributes
  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.

/nw:networks/nw:network/nt:link/l3t:l3-link-attributes/sr
  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:

nw:network-types/l3t:l3-unicast-topology/sr-mpls
  Unauthorized access to this subtree can disclose the SR topology type.

/nw:networks/nw:network/l3t:l3-topology-attributes/sr
Unauthorized access to this subtree can disclose the topology-wide configurations, including the SRGB (Segment Routing Global Block).

/nw:networks/nw:network/nw:node/l3t:l3-node-attributes
Unauthorized access to this subtree can disclose the operational state information of the SR nodes.

/nw:networks/nw:network/nt:link/l3t:l3-link-attributes/sr
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


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 corresponding NMDA model, the YANG tree of the companion module is not depicted separately.

A.1. SR Topology State Module

```Yang
<CODE BEGINS> file "ietf-sr-topology-state@2019-06-28.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 2019-06-28 {
   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;
  }
}

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.


```json
{
   "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": "$203.0.113.1/32$",
         ]
      }
   }
   }
}
```
"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-13-unicast-topology:l3-termination-point-attributes": {  
      "unnumbered-id": 201  
    }  
  },  
  {  
    "tp-id": "2-1-1",  
    "ietf-13-unicast-topology:l3-termination-point-attributes": {  
      "unnumbered-id": 211  
    }  
  },  
  {  
    "tp-id": "2-3-1",  
    "ietf-13-unicast-topology:l3-termination-point-attributes": {  
      "unnumbered-id": 231  
    }  
  }  
],  
"ietf-13-unicast-topology:l3-node-attributes": {  
  "router-id": ["203.0.113.2"],
"prefix": [
    
    ],

"ietf-sr-topology:sr": {
    "start-sid": 102,
    "range": 1,
    "is-local": false,
    "is-node": true
  }
]

"srgb": [
  {
    "lower-bound": 16000,
    "upper-bound": 23999
  }
]

"srlb": [
  {
    "lower-bound": 15000,
    "upper-bound": 15999
  }
]

"node-id": "D3",

"ietf-network-topology:termination-point": [
  
  ],

"ietf-l3-unicast-topology:l3-termination-point-attributes": {
  "unnumbered-id": 311
}

"tp-id": "3-1-1",

"ietf-l3-unicast-topology:l3-termination-point-attributes": {
  "unnumbered-id": 321
}

"tp-id": "3-2-1",

"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"
    },
  }
]
"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
    }
  }
}
"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
Futurewei
EMail: igor.bryskin@futurewei.com

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

Tarek Saad
Juniper Networks
EMail: tsaad@juniper.net

Himanshu Shah
Ciena
EMail: hshah@ciena.com

Stephane Litkowski
Orange Business Service
EMail: stephane.litkowski@orange.com
A YANG Data Model for Traffic Engineering Tunnels and Interfaces
draft-ietf-teas-yang-te-21

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.
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 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 .................................................. 3
   1.1. Terminology .............................................. 3
   1.2. Prefixes in Data Node Names ............................. 3
   1.3. TE Technology Models .................................... 4
   1.4. State Data Organization ................................ 4
2. Model Overview ............................................... 4
   2.1. Module(s) Relationship .................................. 5
   2.2. Design Considerations ................................... 7
   2.3. Model Tree Diagram ..................................... 7
3. Model Organization ........................................... 48
   3.1. Global Configuration and State Data .................... 49
   3.2. Interfaces Configuration and State Data ................ 50
   3.3. Tunnels Configuration and State Data ................... 51
      3.3.1. Tunnel Compute-Only Mode ........................... 51
      3.3.2. Tunnel Hierarchical Link Endpoint .................. 52
   3.4. TE LSPs State Data ...................................... 52
   3.5. Global RPC Data .......................................... 52
   3.6. Interface RPC Data ....................................... 52
   3.7. Tunnel RPC Data .......................................... 52
4. TE Generic and Helper YANG Modules .......................... 53
5. IANA Considerations ......................................... 99
6. Security Considerations ...................................... 99
7. Acknowledgement ............................................. 100
8. Contributors ................................................ 100
9. References .................................................. 101
   9.1. Normative References ................................... 101
   9.2. Informative References ................................. 104
Authors’ Addresses ............................................. 104
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.
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.

```yangle
module: ietf-te
  +--rw te!
    +--rw globals
      +--rw named-admin-groups
        +--rw named-admin-group* [name] [te-types:named-admin-groups]?
          +--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
```
++-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 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
                        +--rw (technology)?
                            +--:(generic)
                                +--rw generic?
                                    rt-types:generalized-label
                            ++-rw direction?
                                te-label-direction
            +--:(srlg)
```yang
++--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
        ++--:(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
```
precise explanation of the YANG data model related to different network configurations and optimizations, including parameters such as tiebreaker, objective function, preference, bandwidth, link protection, setup priority, hold priority, signaling type, path metric bounds, path affinities, disjointness, explicit route objects, and node hops. The document provides a detailed view of the TE YANG data model, which is essential for network engineers to understand and implement advanced network configurations.
++-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 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
++-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

```yang
++-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
                    |     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
                    |     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
  ++-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)?
        ++-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
<snip>
---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)
    +---rw (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
<snip>
```yang
---(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
```
Internet-Draft             TE YANG Data Model                 April 2019

| +--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 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?
    ++--ro 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
Internet-Draft             TE YANG Data Model                 April 2019

+++: (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
+++: (sr\(1\))
  ++-rw srlg
  ++-rw srlg? uint32
++-rw explicit-route-in\(c\)lude-objects
  +++-rw route-object-in\(c\)lude-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 (technology)?
  ++-:(generic)
  ++-rw generic?
      rt-types:generalized-label
te-label-direction
---rw direction?
  te-label-direction
---rw label-end
  ---rw te-label
  ++-rw (technology)?
    ++-:(generic)
    ++-rw generic?
      rt-types:generalized-label
te-label-direction
---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
---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 k-index                         uint8      |
|   |   | path-properties                    |
|   |   |   | path-metric* [metric-type]         identityref|
|   |   |   | accumulative-value?                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  |
|   |   | srlgs-lists                        |
|   |   |   | srlgs-list* [usage]                |
|   |   |   |   | usage                          identityref|
|   |   |   |   | values*                         srlg   |
|   |   | names*                             string   |
|   |   | route-objects                      |
|   |   |   | computed-route-object* [index]    |
|   |   |   |       | index                         uint32   |
|   |   |       | (type)?                          |
|   |   |   |   | numbered-node-hop               |
|   |   |   |       | node-id                         te-node-id|
|   |   |   |       | hop-type?                        te-hop-type|
|   |   |   | numbered-link-hop                 |
|   |   |       | 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 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)?
               ---:(numbered-node-hop)
| ---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  

Internet-Draft             TE YANG Data Model                 April 2019

+--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
      +--rw (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:te-advertisements-state
            +--ro te-dev:flood-interval?     uint32
            +--ro te-dev:last-flooded-time?  uint32

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:
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 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 show 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:tunnels/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:tunnels/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:tunnels/te:tunnel/" + 
        "te:p2p-primary-paths/te:p2p-primary-path/te:name";
    }
    type leafref {
      path "te:tunnels/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 computation 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;
   }
}
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-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 "../../../global/" + "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 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 external 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 external 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)";
}

grouping protection-restoration-properties {

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-pscl;
        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-pscl;
  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;
description "Association extended ID";
reference "RFC4872";
}
}

/* 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 tunnnel";
    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";
            }
          }
        }
      }
    }
  }
}
Figure 7: TE generic YANG module

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
/* 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>
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";
}

/**
* TE LSP device state grouping
*/
grouping lsps-device-state {
  description "TE LSP device state grouping";
  container lsp-timers {
    when "./*: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
    }
}
"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 invalidtion 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 parameters */
    uses te-attributes;
  }
}

/**
 * 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";
  }
}

case named-admin-groups {
  list named-admin-groups {
    if-feature te-types:extended-admin-groups;
    if-feature te-types:named-extended-admin-groups;
    key named-admin-group;
    description "A list of named admin-group entries";
    leaf named-admin-group {
      type leafref {
        path "../../../../te:globals/" +
        "te:named-admin-groups/te:named-admin-group/" +
        "te:name";
      }
      description "A named admin-group entry";
    }
  }
}

/* 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";
          }
        }
      }
    }
  }
}
case named-srlgs {
  list named-srlgs {
    if-feature te-types:named-srlg-groups;
    key named-srlg;
    description "A list of named SRLG entries";
    leaf named-srlg {
      type leafref {
        path "../../../../te:globals/te:named-srlgs/te:named-srlg/te:name";
      }
      description "A named SRLG entry";
    }
  }
}

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 >= 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 >= 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-down 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;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
    "/te:p2p-secondary-path/te:lsps/te:lsp" {
    description
    "LSP device dependent augmentation";
    uses lsps-device-state;
}

augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
    "/te:p2p-primary-path/te:lsps/te:lsp" {
    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:

"/teGlobals": 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
Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Igor Bryskin
Huawei Technologies

Email: Igor.Bryskin@huawei.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]
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
Internet-Draft       MPLS TE Tunnels YANG Data Model       February 2019

+--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 /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 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: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
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";

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 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 ietf-te-types {
    prefix te-types;
    reference "draft-ietf-teas-yang-te-types: A YANG Data Model for Common Traffic Engineering 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.
/* 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;
default false;
description
"Enables MPLS auto-bandwidth on the LSP";
}
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
"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].

```yaml
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
Applicability of ACTN to Support 5G Transport
draft-lee-teas-actn-5g-transport-00

Abstract

This draft is aimed to provide an applicability of Abstraction and Control of Traffic Engineered (TE) Networks (ACTN) for an end-to-end service assurance mechanism for 5G transport network. ACTN is an IETF standard architecture enabling virtual network operations to control and manage large-scale multi-domain, multi-layer and multi-vendor TE networks, so as to facilitate network programmability, automation, efficient resource sharing. 3GPP 5G requirements calls for Network Slicing support for various use cases such as enhanced mobile broadband (eMBB), massive machine-type communications (mMTC) and ultra-reliable and low latency communications (URLLC). In order to support these new requirements over multiple transport networks for 5G transport, the current 3GPP 5G architecture needs to support dynamic instantiation of end-to-end paths that assure service assurance and performance guarantee for traffic classes characterized by network slicing.

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 January 8, 2020.

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.

Table of Contents

1. Introduction.................................................3
1.1. Requirements Language.....................................4
2. IETF ACTN Virtual Network Slicing Service Model..............4
3. 3GPP 5G Network Architecture................................5
4. Transport Network Provisioning................................7
  4.1. Mobile Transport Network Context..........................7
  4.2. Transport Provisioning....................................8
5. Federated Orchestration and Controller Functions.............9
7. Scalability Considerations..................................11
8. Security Considerations....................................11
9. IANA Considerations..........................................12
10. Acknowledgements..........................................12
11. References................................................12
  11.1. Normative References...................................12
  11.2. Informative References................................13
12. Contributors..............................................13
Authors’ Addresses.............................................13
1. Introduction

ACTN framework defines the requirements, use cases, and an SDN-based architecture, relying on the concepts of network and service abstraction, detaching the network and service control from the underlying data plane. ACTN architecture encompasses Provisioning Network Controllers (PNCs), responsible for specific technology and administrative domains, orchestrated by Multi-Domain Service Coordinator (MDSC), which, in turn, enables underlay transport resources to be abstracted and virtual network instances to be allocated to customers and applications, under the control of a Customer Network Controller (CNC) [RFC8453].

A network slice is defined by 3GPP as an end-to-end logical network comprising a set of managed resources and network functions [3GPP TS 28.531]. Its definition and deployment starts from the RAN (Radio Access Network) and packet core, but in order to guarantee end to end SLAs (Service Level Agreements) and KPIs (Key Performance Indicators) especially for applications that require strict latency and bandwidth guarantee, the transport network also plays an important role and needs to be sliced as part of services bound to the different slices.

However, it is not easy for mobile network clients to interface directly with transport networks [Transport-Slicing]. Current GSMA guidelines for interconnection with transport networks [IR.34-GSMA] provide an application mapping into DSCP. However, apart from problems with classification of encrypted packets, these recommendations do not take into consideration other aspects in slicing like isolation, protection and replication. For example, during a PDU session setup the 3GPP control plane selects a 3GPP slice, 5QI (QoS parameters) and programs the user plane (gNB, UPF). This 3GPP slice and QoS firstly needs to have a corresponding mapping in the transport network segment(s) between the 3GPP user plane functions (N 3GPP Slices: M Transport). Secondly, there needs to be a mechanism for carrying the meta-data corresponding to the mapping in IP packet header so that the transport network can grant the service level provisioned.

ACTN has been driving SDN standardization in IETF in the TEAS (Traffic Engineering and Signaling) WG with the emphasis of
providing the desired customer interfaces that enable dynamic and automatic transport network slice instantiation and its life cycle operation [VN-Model], [Transport-Slicing].

This draft presents an extended ACTN architecture with 3GPP 5G transport architecture in order to provide a novel approach for an end-to-end service assurance mechanism to meet 3GPP 5G requirements for support of enhanced mobile broadband (eMBB) and for new ‘use cases’ that require massive machine-type communications (mMTC) and ultra-reliable and low latency communications (URLLC). In addition, this draft addresses requirements for transport network provisioning function requirements and data plane network programming to support end-to-end service assurance mechanism.

1.1. 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].

2. IETF ACTN Virtual Network Slicing Service Model

IETF ACTN VN model [VN-Model] discusses customer initiated virtual network slicing data model in which customer can control their virtual network slice to fit their needs. This model fulfills the key requirement: the ability for the customer to define and convey their virtual networks without having to understand transport network details [VN-Model]. This is for CNC (Customer Network Controller) - MDSC (Multi-domain Service Coordinator) Interface (CMI) of ACTN, as shown in Figure 1. This model describes VN YANG model for customer access points, virtual network access points, Virtual Network (VN) identifiers, its VN-members, any constraints and policy customer cares for with respect to its VNs. Figure 1 shows the process of VN creation in the context of ACTN architecture.
Figure 1 [VN-Model] shows that VN Slicing Service model enables customer to create its VN without having to know the transport underlay details and to indicate its end-points with constraints (e.g., bandwidth, latency, load-balancing, protection, etc.) per VN or VN-member level. This model facilitates customer-driven dynamic life-cycle VN service operation.

The CMI plays an important role interfacing 5G 3GPP mobile network with transport networks. From a context of 5G transport network architecture, the CNC is the entity that is responsible for 3GPP access network coordination with transport networks. This entity is referred to as Traffic Provisioning Manager (TPM) for 3GPP/5G context. Sections 3 and 4 discuss TPM function in details.

3. 3GPP 5G Network Architecture

Mobile network backhauls in the past have used static configuration and provisioning of routers for traffic engineering (TE). These estimates may be revised and TE is configured periodically based on demand and other performance criteria—however, this process takes a long time (in the order of weeks or
months), thus may not be suitable for dynamically changing context such as 5G mobile network.

In 5G systems [3GPP TS 23.501], [3GPP TS 23.502] with a large range of services, low latency paths and mobility, the demand estimate varies much more dynamically (in the order of several minutes in the worst cases). Backhaul networks that provide capabilities to reprogram routers and switches to meet the new demand profile are needed.

In addition to the configuration and provisioning of traffic engineered paths between mobile and transport network providers, there is the question of how to enforce policies for slices, QoS across multiple transport network domains in mobile network and transport network. Each transport domain may employ different data plane technologies such as IP, MPLS, SR-MPLS, SRV6, OTN/WDM, etc. From an end-to-end 5G transport network perspective, it is paramount to ensure predictable and consistent service quality across all domains.

Figure 2 shows an enhanced 5G transport network architecture with an overview of the TPM function. The TPM is deployed in each of the two domains/sites (Domain 1/Site 1, Domain 2/Site 2) and interfaces with other mobile network functions (e.g., Session Management Function (SMF), SDN Controller (SDN-C), etc.) while providing interfaces to transport network orchestrator (i.e., MDSC). Note that TPM is a new function to be added and implemented in the current 3GPP architecture and this should be addressed in 3GPP. Detailed description of the TPM is in section 4.

Figure 2 shows three segments/domains for 5G transport network.

. N3 segment/domain between Next Generation NodeB (gNB) and User Plane Function (UPF) - Uplink Classifier (ULCL) is over the transport network at that site (Data Center/Central Office).

. N9 mobile connection transport, there are three transport segments/domains - the transport at each mobile network site (1, 2) and the backhaul network in between.

. N6 transport segment between UPF - PDU Session Identifier (PSA) and Application Servers (AS) is over the transport network at that site (Data Center/Central Office).

Figure 2. Enhanced 5G Transport Network Architecture

The N3, N9 and N6 transport segments outlined in the figure are exemplary. For instance, gNB itself may need transport network when DU (Distribution Unit) and CU (Central Unit) are separated.

4. Transport Network Provisioning

4.1. Mobile Transport Network Context

The TPM in Domain 1 in Figure 2 is the initiator of the e2e network slice policy as it would estimate traffic matrix and determine service quality for each traffic class coupled with network slice requirement. This policy is referred to as Multi Transport Network Context (MTNC) and identified with MTNC Identifier. The MTNC Identifier is allocated for each traffic class.

The MTNC represents a transport network slice, QoS configuration for a transport path/VN between two 3GPP user plane functions (e.g., between gNB and UPF and between UPF-ULCL and UPF-PSA) and between UPF-PSA and Application Servers (AS). The MTNC include a set of requirements, such as quality of service (QoS) requirements, class of service (CoS), a resilience requirement, and/or an isolation requirement, and so on, according to which transport resources of a transport network are provisioned for routing traffic between two service end points.
The MTNC identifier is generated by the TPM to be unique for each path and per traffic class (including QoS and slice aspects). Thus, there may be more than one MTNC identifier for the same QoS and path if there is a need to provide isolation (slice) of the traffic. It should be noted that MTNC identifiers are per class/path and not per user session (nor is it per data path entity). The MTNC identifiers are configured by the TPM to be unique within a provisioning domain.

4.2. Transport Provisioning

As introduced previously, from a context of 5G transport network architecture, the TPM (a type of CNC) is the entity that is responsible for 3GPP access network coordination with the backhaul transport network. The TPM is the requester of VNs and collaborates with the MDSC to form a closed feedback loop with regard to traffic class associated with each VN, which in turn maps with network slice requirements. Thus, the TPM plays a central role from an orchestration point of view interacting with transport network’s orchestration (i.e., MDSC) and with other TPMs in other domains.

The Transport Path Manager (TPM) is a logical entity that can be part of Network Slice Selection Management Function (NSSMF) in the 3GPP management plane [TS.28.533-3GPP]. The TPM may use network configuration, policies, history, heuristics or some combination of these to derive traffic estimates that the TPM would use. How these estimates are derived and the precise 3GPP entity that hosts the TPM functionality are not in the scope of this document. The focus here is only in terms of how the TPM and SDN-C are programmed given that slice and QoS characteristics across a transport path can be represented by a Mobile Transport Network Context (MTNC) identifier.

TPM creates the MTNC identifier provisioned to control and user plane functions in the 3GPP domain. Once the MTNC identifier is created by the TPM, the TPM then requests the SDN-C in the transport network to provision paths in the transport domain based on the MTNC identifier. Federated orchestration and controller aspects in relation to TPM are discussed in Section 5. Detailed
mechanisms for programming the MTNC identifier across 3GPP control and user plane should be part of the 3GPP specifications.

5. Federated Orchestration and Controller Functions

The TPM is a type of the CNC as depicted in Figure 1. The TPMs and the MDSC form a federated orchestration relationship to each other in order to collaborate network slice policy and implement the negotiated network slice policy to its domain network, respectively.

The SDN controllers of each domain are responsible to create per class domain paths/VNs meeting the MTNC requirements. Once per class domain paths/VNs are created using ACTN VN model, the SDN controller would need to program the domain ingress router/network switch to populate the routing instruction so that the data packets associated with the MTNC identifier would be routed to the pre-established paths/VNs for the MTNC identifier.

[ACTN-PM] discusses models that allow customers (e.g., TPM) 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. This model can be implemented as a way to support network automation by forming a close-loop relationship between controller entities (e.g., TPM - MDSC, TPM - SDN controller, etc.)

6. Network Programming Function Over Data Plane

There is a need to carry the MTNC identifier in data packets:

- Slices and QoS classes in the service domain do not have a 1:1 correspondence between the 3GPP domain and the transport domain. Some meta-data or token to associate information provisioned across 3GPP-transport domains needs to be carried in the data packets that need to get specific treatment in the transport domain.

- The MTNC identifier (which is meta-data) that is carried in the data packet header should be at the granularity of the provisioning for services between the 3GPP and transport
domains. Specifically, the service is provided by the transport domain and the meta-data should be used in the transport domain to classify packets and provide the services agreed to.

Protocol extensions to carry the above policy meta-data across connection segments between 3GPP functions (N3, N9) and also across 3GPP - to external system (N6, e.g., to application server)

In order to support the data plane programming with MTNC identifier, the TPM would need to propagate MTNC identifiers within the 3GPP control and user plane. These 3GPP control and user plane mechanisms should be standardized as part of 3GPP specifications.

Figure 2 shows that for N3, the data packets are "stamped" with the proper MTNC identifier by the gNBs via UDP header encapsulation mechanism as an illustration. As for N9 and N6, the UPFs would need to stamp the data packets with the same MTNC identifier for the next domain. For each domain, all the packets identified by the MTNC identifier will be routed to the pre-established paths/VNs to ensure the proper level of service performance for the traffic class associated with the MTNC identifier.

When the 3GPP user plane function (gNB, UPF) and transport provider edge are on different nodes, the edge router needs to have the means by which to classify the PDU packet. IP header fields such as DSCP (DiffServ Code Point) or the IPv6 Flow Label do not satisfy the requirement as they are not immutable. GTP-U [TS.29.281-3GPP] extension headers are not the best option either as the extension fields are chained and would potentially require significant processing by the transport edge router. Further, GTP-U extension fields carry 3GPP information between two 3GPP network functions and is not meant to carry information to be processed by the IP transport plane.

The provisioning mechanisms here expect that the MTNC identifier is carried in the IP packet header (PDU session data packet). This MTNC identifier is used to classify the PDU packet at the transport edge router. The MTNC identifier should be carried in some IP header field and should not be modified on path. Transport edge routers should only inspect the MTNC identifier for each packet and derive the class of transport service that should be provided (e.g., with MPLS or segment routes).
Different options for carrying the MTNC identifier in the IP data packet include SRv6 [I-D.ietf-spring-segment-routing] and GUE [I-D.ietf-intarea-gue-extensions]. The SRv6 is an underlay where the MTNC identifier can be encoded into Segment Routing Headers (SRH) that are then used to forward the PDU packet in the transport domain. The GUE headers, on the other hand, constitute an overlay mechanism where the MTNC identifier can also be encapsulated in the UDP extension header fields. A transport network like MPLS would inspect the MTNC header field in GUE and point to its already programmed label switched path. There are various trade-offs in terms of packet overhead, support in IPv4 and IPv6 networks as well as working across legacy and evolving transport networks that need to be considered. These considerations will be addressed in other future drafts.

7. Scalability Considerations

Since the MTNC-IDs represent QoS and slice of the service domain that is mapped to a transport domain slice for a path between two network functions (NF), there are multiple flows that get mapped to a single such transport slice. The number of transport slices to be provisioned scales well as it is related to the number of sites \((N(N-1)/2) \times Q\) for \(N\) number of sites, \(Q\) classes of service). For example, if there are 25 sites and 3 classes of service, the number of paths provisioned will at most be 900, while the number of PDN connection flows handled over those connections can be well over a million. As the number of transport paths setup is a few orders lower than the number of connections/flows that are handled, these mechanisms scale extremely well compared to setting this up per PDN connection.

8. Security Considerations

From a security and reliability perspective, ACTN may encounter many risks such as malicious attack and rogue elements attempting to connect to various ACTN components. Furthermore, some ACTN components represent a single point of failure and threat vector and must also manage policy conflicts and eavesdropping of communication between different ACTN components.

All protocols used to realize the ACTN framework should have rich security features, and customer, application and network data should be stored in encrypted data stores. Additional security risks may still exist. Therefore, discussion and applicability of specific security functions and protocols will be better described in documents that are use case and environment specific.

The CMI will likely be an external protocol interface. Suitable authentication and authorization of each CNC connecting to the MDSC will be required; especially, as these are likely to be implemented by different organizations and on separate functional nodes. Use of the AAA-based mechanisms would also provide role-based authorization methods so that only authorized CNC’s may access the different functions of the MDSC.

Where the MDSC must interact with multiple (distributed) PNCs, a PKI-based mechanism is suggested, such as building a TLS or HTTPS connection between the MDSC and PNCs, to ensure trust between the physical network layer control components and the MDSC. Trust anchors for the PKI can be configured to use a smaller (and potentially non-intersecting) set of trusted Certificate Authorities (CAs) than in the Web PKI. Which MDSC the PNC exports topology information to, and the level of detail (full or abstracted), should also be authenticated, and specific access restrictions and topology views should be configurable and/or policy based.

9. IANA Considerations

This document has no IANA actions.

10. Acknowledgements

The authors thank James Guichard for useful discussions and his suggestions for this work.

11. References

11.1. Normative References
11.2. Informative References


12. Contributors

Authors' Addresses

Y. Lee
Futurewei Technologies
Email: younglee.tx@gmail.com

John Kaippallimalil
Futurewei Technologies
Applicability of ACTN to Support Packet and Optical Integration
draft-lee-teas-actn-poi-applicability-00

Abstract

This document outlines the applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Packet & Optical Integration (POI). It also identifies a number of deployment scenarios to support L3VPN and L2VPN in operator’s networks and provides implementation guidelines.

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 20, 2019.

Copyright Notice

Lee, et al. Expires February 2019
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.

Table of Contents

1. Introduction.................................................3
   1.1. Requirements Language.....................................3
2. POI with L2/L3VPN Service Under Single Network Operator Control ..........................................................3
   2.1. L2/L3VPN/VN Service Request by the Customer ...............5
   2.2. Service and Network Orchestration............................7
   2.3. IP/MPLS Domain Controller and NE Functions..............10
       2.3.1. Scenario A: Shared Tunnel Selection...................10
           2.3.1.1. Domain Tunnel Selection........................11
           2.3.1.2. VPN/VRF Provisioning for L3VPN.................12
           2.3.1.3. VSI Provisioning for L2VPN.....................13
           2.3.1.4. Inter-domain Links Update......................13
           2.3.1.5. End-to-end Tunnel Management...................13
   2.3.2. Scenario B: Isolated VN/Tunnel Establishment...........14
   2.4. Optical Domain Controller and NE Functions...............14
   2.5. Orchestrator-Controllers-NEs Communication Protocol Flows 16
3. POI with VN Recursion Under Multiple Network Operators Control ..........................................................17
   3.1. Service Request Process between Multiple Operators.......19
   3.2. Service/network Orchestration of Operator 2...............19
4. Security Considerations.....................................20
5. IANA Considerations..........................................21
6. Acknowledgements............................................21
7. References..................................................21
   7.1. Normative References....................................21
   7.2. Informative References...................................21
8. Contributors................................................22
   Authors’ Addresses............................................22
1. Introduction

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 so that, for example, a link in the customer’s network is constructed from a path or collection of paths in the underlying networks [RFC8453].

This document outlines the applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Packet and Optical Integration. It also identifies a number of deployment scenarios to support POI in operator’s networks and provides implementation guidelines.

1.1. 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].

2. POI with L2/L3VPN Service Under Single Network Operator Control

This section provides a number of deployment scenarios for packet and optical integration (POI). Specifically, this section provides a deployment scenario in which ACTN hierarchy is deployed to control a multi-layer and multi-domain network via two IP/MPLS PNCs and two Optical PNCs with coordination with L-MDSC. This scenario is in the context of an upper layer service configuration (e.g. L3VPN) across two AS domains which are transported by two transport underlay domains (e.g. OTN).

The provisioning of the L3VPN service is outside ACTN scope but it is worth showing how the L3VPN service provisioning is integrated for the end-to-end service fulfilment in ACTN context. An example of service configuration function in the Service/Network Orchestrator is discussed in [bess-l3vpn].

Figure 1 shows an ACTN POI Reference Architecture where it shows ACTN components as well as non-ACTN components that are necessary for the end-to-end service fulfilment. Both IP/MPLS and Optical Networks are multi-domain. Each IP/MPLS domain network is controlled by its’ domain controller and all the optical domains are controlled by a hierarchy of optical domain controllers. The L-MDSC function of the optical domain controllers provides an abstract view of the whole optical network to the Service/Network Orchestrator. It is assumed that all these components of the
network belong to one single network operator domain under the control of the service/network orchestrator.

Figure 1. ACTN POI Reference Architecture

Figure 1 shows ACTN POI Reference Architecture where it depicts:

- CMI (CNC-MDSC Interface) interfacing CNC with MDSC function in the Service/Network Orchestrator. This is where TE & Service Mapping [TSM] and either ACTN VN [ACTN-VN] or TE-topology [TE-Topo] model is exchanged over CMI.
. Customer Service Model Interface: Non-ACTN interface in the Customer Portal interfacing Service/Network Orchestrator’s Service Configuration Function. This is the interface where L3SM information is exchanged.

. MPI (MDSC-PNC Interface) interfacing IP/MPLS Domain Controllers and Optical Domain Controllers.

. Service Configuration Interface: Non-ACTN interface in Service/Network Orchestrator interfacing with the IP/MPLS Domain Controllers to coordinate L2/L3VPN multi-domain service configuration. This is where service specific information such as VPN, VPN binding policy (e.g., new underlay tunnel creation for isolation), etc. are conveyed.

. SBI (South Bound Interface): Non-ACTN interface in the domain controller interfacing network elements in the domain.

Please note that MPI and Service Configuration Interface can be implemented as the same interface with the two different capabilities. The split is just functional but doesn’t have to be also logical.

The following sections are provided to describe key functions that are necessary for the vertical as well as horizontal end-to-end service fulfilment of POI.

2.1. L2/L3VPN/VN Service Request by the Customer

A customer can request L3VPN services with TE requirements using ACTN CMI models (i.e., ACTN VN YANG, TE & Service Mapping YANG) and non-ACTN customer service models such as L2SM/L3SM YANG together. Figure 2 shows detailed control flow between customer and service/network orchestrator to instantiate L2/L3VPN/VN service request.
ACTN VN YANG provides VN Service configuration, as specified in [ACTN-VN].

- It provides the profile of VN in terms of VN members, each of which corresponds to an edge-to-edge link between customer end-points (VNAPs). It also provides the mappings between the VNAPs with the LTPs and between the connectivity matrix with the VN member from which the associated traffic matrix (e.g., bandwidth, latency, protection level, etc.) of VN member is expressed (i.e., via the TE-topology’s connectivity matrix).

- The model also provides VN-level preference information (e.g., VN member diversity) and VN-level admin-status and operational-status.

L2SM YANG [RFC8466] provides all L2VPN service configuration and site information from a customer/service point of view.
Internet-Draft        ACTN Applicability to POI               June 2019

L3SM YANG [RFC8299] provides all L3VPN service configuration and site information from a customer/service point of view.

The TE & Service Mapping YANG model [TE & Service] provides TE-service mapping as well as site mapping.

- TE-service mapping provides the mapping of L3VPN instance from [RFC8299] with the corresponding ACTN VN instance.
- The TE-service mapping also provides the service mapping requirement type as to how each L2/L3VPN/VN instance is created with respect to the underlay TE tunnels (e.g., whether the L3VPN requires a new and isolated set of TE underlay tunnels or not, etc.). See Section 2.2 for detailed discussion on the mapping requirement types.
- Site mapping provides the site reference information across L2/L3VPN Site ID, ACTN VN Access Point ID, and the LTP of the access link.

2.2. Service and Network Orchestration

The Service/Network orchestrator shown in Figure 1 interfaces the customer and decouples the ACTN MDSC functions from the customer service configuration functions.

An implementation can choose to split the Service/Network orchestration functions, as described in [RFC8309] and in section 4.2 of [RFC8453], between a top-level Service Orchestrator interfacing the customer and two low-level Network Orchestrators, one controlling a multi-domain IP/MPLS network and the other controlling the Optical networks.

Another implementation can choose to combine the L-MDSC functions of the Optical hierarchical controller, providing multi-domain coordination of the Optical network together with the MDSC functions in the Service/Network orchestrator.

Without loss of generality, this assumes that the service/network orchestrator as depicted in Figure 1 would include all the required functionalities as in a hierarchical orchestration case.

One of the important service functions the Service/Network orchestrator performs is to identify which TE Tunnels should carry the L3VPN traffic (from TE & Service Mapping Model) and to relay this information to the IP/MPLS domain controllers, via non-ACTN
interface, to ensure proper IP/VRF forwarding table be populated according to the TE binding requirement for the L3VPN.

[Editor’s Note: What mechanism would convey on the interface to the IP/MPLS domain controllers as well as on the SBI (between IP/MPLS domain controllers and IP/MPLS PE routers) the TE binding policy dynamically for the L3VPN? Typically, VRF is the function of the device that participate MP-BGP in MPLS VPN. With current MP-BGP implementation in MPLS VPN, the VRF’s BGP next hop is the destination PE and the mapping to a tunnel (either an LDP or a BGP tunnel) toward the destination PE is done by automatically without any configuration. It is to be determined the impact on the PE VRF operation when the tunnel is an optical bypass tunnel which does not participate either LDP or BGP.

Figure 3 shows service/network orchestrator interactions with various domain controllers to instantiate tunnel provisioning as well as service configuration.

![Diagram of service and network orchestration process]

Figure 3. Service and Network Orchestration Process

TE binding requirement types [TE-service mapping] are:

1. Hard Isolation with deterministic latency: Customer would request an L3VPN service [RFC8299] using a set of TE
Tunnels with a deterministic latency requirement and that cannot be not shared with other L3VPN services nor compete for bandwidth with other Tunnels.

2. Hard Isolation: This is similar to the above case without deterministic latency requirements.

3. Soft Isolation: Customer would request an L3VPN service using a set of MPLS-TE tunnel which cannot be shared with other L3VPN services.

4. Sharing: Customer would accept sharing the MPLS-TE Tunnels supporting its L3VPN service with other services.

For the first three types, there could be additional TE binding requirements with respect to different VN members of the same VN associated with an L3VPN service. For the first two cases, VN members can be hard-isolated, soft-isolated, or shared. For the third case, VN members can be soft-isolated or shared.

When "Hard Isolation with or w/o deterministic latency" (i.e., the first and the second type) TE binding requirement is applied for a L3VPN, a new optical layer tunnel has to be created (Step 1 in Figure 3). This operation requires the following control level mechanisms as follows:

- The MDSC function of the Service/Network Orchestrator identifies only the domains in the IP/MPLS layer in which the VPN needs to be forwarded.

- Once the IP/MPLS layer domains are determined, the MDSC function of the Service/Network Orchestrator needs to identify the set of optical ingress and egress points of the underlay optical tunnels providing connectivity between the IP/MPLS layer domains.

- Once both IP/MPLS layers and optical layer are determined, the MDSC needs to identify the inter-layer peering points in both IP/MPLS domains as well as the optical domain(s). This implies that the L3VPN traffic will be forwarded to an MPLS-TE tunnel that starts at the ingress PE (in one IP/MPLS domain) and terminates at the egress PE (in another IP/MPLS domain) via a dedicated underlay optical tunnel.

- The MDSC function of the Service/Network Orchestrator needs to first request the optical L-MDSC to instantiate an optical
tunnel for the optical ingress and egress. This is referred to as optical tunnel creation (Step 1 in Figure 3). Note that it is L-MDSC responsibility to perform multi-domain optical coordination with its underlying optical PNCs, for setting up a multi-domain optical tunnel.

Once the optical tunnel is established, then the MDSC function of the Service/Network Orchestrator needs to coordinate with the PNC functions of the IP/MPLS Domain Controllers (under which the ingress and egress PEs belong) the setup of a multi-domain MPLS-TE Tunnel, between the ingress and egress PEs. This setup is carried by the created underlay optical tunnel (Step 2 in Figure 3).

It is the responsibility of the Service Configuration Function of the Service/Network Orchestrator to identify interfaces/labels on both ingress and egress PEs and to convey this information to both the IP/MPLS Domain Controllers (under which the ingress and egress PEs belong) for proper configuration of the L3VPN (BGP and VRF function of the PEs) in their domain networks (Step 3 in Figure 3).

2.3. IP/MPLS Domain Controller and NE Functions

IP/MPLS networks are assumed to have multiple domains and each domain is controlled by IP/MPLS domain controller in which the ACTN PNC functions and non-ACTN service functions are performed by the IP/MPLS domain controller.

Among the functions of the IP/MPLS domain controller are VPN service aspect provisioning such as VRF control and management for VPN services, etc. It is assumed that BGP is running in the inter-domain IP/MPLS networks for L2/L3VPN and that the IP/MPLS domain controller is also responsible for configuring the BGP speakers within its control domain if necessary.

Depending on the TE binding requirement types discussed in Section 2.2., there are two possible deployment scenarios.

2.3.1. Scenario A: Shared Tunnel Selection

When the L2/L3VPN does not require isolation (either hard or soft), it can select an existing MPLS-TE and Optical tunnel between ingress and egress PE, without creating any new TE tunnels. Figure 4 shows this scenario.
How VPN is disseminated across the network is out of the scope of this document. We assume that MP-BGP is running in IP/MPLS networks and VPN is made known to ABSRs and PEs by each IP/MPLS domain controllers. See RFC 4364 [RFC4364] for detailed descriptions on how MP-BGP works.

There are several functions IP/MPLS domain controllers need to provide in order to facilitate tunnel selection for the VPN in both domain level and end-to-end level.

2.3.1.1. Domain Tunnel Selection

Each domain IP/MPLS controller is responsible for selecting its domain level tunnel for the L3VPN. First it needs to determine which existing tunnels would fit for the L2/L3VPN requirements allotted to the domain by the Service/Network Orchestrator (e.g., tunnel binding, bandwidth, latency, etc.). If there are existing tunnels that are feasible to satisfy the L3VPN requirements, the IP/MPLS domain controller selects the optimal tunnel from the candidate pool. Otherwise, an MPLS tunnel with modified bandwidth or a new MPLS Tunnel needs to be setup. Note that with no isolation requirement for the L3VPN, existing MPLS tunnel can be selected. With soft isolation requirement for the L3VPN, an optical tunnel can be shared with other L2/L3VPN services while...
with hard isolation requirement for the L2/L3VPN, a dedicated MPLS-TE and a dedicated optical tunnel MUST be provisioned for the L2/L3VPN.

2.3.1.2. VPN/VRF Provisioning for L3VPN

Once the domain level tunnel is selected for a domain, the Service Function of the IP/MPLS domain controller maps the L3VPN to the selected MPLS-TE tunnel and assigns a label (e.g., MPLS label) with the PE. Then the PE creates a new entry for the VPN in the VRF forwarding table so that when the VPN packet arrives to the PE, it will be able to direct to the right interface and PUSH the label assigned for the VPN. When the PE forwards a VPN packet, it will push the VPN label signaled by BGP and, in case of option A and B [RFC4364], it will also push the LSP label assigned to the configured MPLS-TE Tunnel to reach the ASBR next hop and forwards the packet to the MPLS next-hop of this MPLS-TE Tunnel.

In case of option C [RFC4364], the PE will push one MPLS LSP label signaled by BGP to reach the destination PE and a second MPLS LSP label assigned to the configured MPLS-TE Tunnel to reach the ASBR next-hop and forward the packet to the MPLS next-hop of this MPLS-TE Tunnel.

With Option C, the ASBR of the first domain interfacing the next domain should keep the VPN label intact to the ASBR of the next domain so that the ASBR in the next domain sees the VPN packets as if they are coming from a CE. With Option B, the VPN label is swapped. With option A, the VPN label is removed.

With Option A and B, the ASBR of the second domain does the same procedure that includes VPN/VRF tunnel mapping and interface/label assignment with the IP/MPLS domain controller. With option A, the ASBR operations are the same as of the PEs. With option B, the ASBR operates with VPN labels so it can see the VPN the traffic belongs to. With option C, the ASBR operates with the end-to-end tunnel labels so it may be not aware of the VPN the traffic belongs to.

This process is repeated in each domain. The PE of the last domain interfacing the destination CE should recognize the VPN label when the VPN packets arrive and thus POP the VPN label and forward the packets to the CE.
2.3.1.3. VSI Provisioning for L2VPN

The VSI provisioning for L2VPN is similar to the VPN/VRF provision for L3VPN. L2VPN service types include:

- Point-to-point Virtual Private Wire Services (VPWSs) that use LDP-signaled Pseudowires or L2TP-signaled Pseudowires [RFC6074];
- Multipoint Virtual Private LAN Services (VPLSs) that use LDP-signaled Pseudowires or L2TP-signaled Pseudowires [RFC6074];
- Multipoint Virtual Private LAN Services (VPLSs) that use a Border Gateway Protocol (BGP) control plane as described in [RFC4761] and [RFC6624];
- IP-Only LAN-Like Services (IPLSs) that are a functional subset of VPLS services [RFC7436];
- BGP MPLS-based Ethernet VPN Services as described in [RFC7432] and [RFC7209];
- Ethernet VPN VPWS specified in [RFC8214] and [RFC7432].

2.3.1.4. Inter-domain Links Update

In order to facilitate inter-domain links for the VPN, we assume that the service/network orchestrator would know the inter-domain link status and its resource information (e.g., bandwidth available, protection/restoration policy, etc.) via some mechanisms (which are beyond the scope of this document). We also assume that the inter-domain links are pre-configured prior to service instantiation.

2.3.1.5. End-to-end Tunnel Management

It is foreseen that the Service/Network orchestrator should control and manage end-to-end tunnels for VPNs per VPN policy.

As discussed in [ACTN-PM], the Orchestrator is responsible to collect domain LSP-level performance monitoring data from domain controllers and to derive and report end-to-end tunnel performance monitoring information to the customer.
2.3.2. Scenario B: Isolated VN/Tunnel Establishment

When the L3VPN requires hard-isolated Tunnel establishment, optical layer tunnel binding with IP/MPLS layer is necessary. As such, the following functions are necessary.

- The IP/MPLS Domain Controller of Domain 1 needs to send the VRF instruction to the PE:

  o To the Ingress PE of AS Domain 1: Configuration for each L3VPN destination IP address (in this case the remote CE’s IP address for the VPN or any customer’s IP addresses reachable through a remote CE) of the associated VPN label assigned by the Egress PE and of the MPLS-TE Tunnel to be used to reach the Egress PE: so that the proper VRF table is populated to forward the VPN traffic to the inter-layer optical interface with the VPN label.

  o The Egress PE, upon the discovery of a new IP address, needs to send the mapping information (i.e., VPN to IP address) to its’ IP/MPLS Domain Controller of Domain 2 which sends, in turn, to the service orchestrator. The service orchestrator would then propagate this mapping information to the IP/MPLS Domain Controller of Domain 1 which sends it, in turn, to the ingress PE so that it may override the VPN/VRF forwarding or VSI forwarding, respectively for L3VPN and L2VPN. As a result, when packets arriving at the ingress PE with that IP destination address, the ingress PE would then forward this packet to the inter-layer optical interface.

[Editor’s Note: in case of hard isolated tunnel required for the VPN, we need to create a separate MPLS TE tunnel and encapsulate the MPLS packets of the MPLS Tunnel into the ODU so that the optical NE would route this MPLS Tunnel to a separate optical tunnel from other tunnels.]

2.4. Optical Domain Controller and NE Functions

Optical network provides the underlay connectivity services to IP/MPLS networks. The multi-domain optical network coordination is performed by the L-MDSC function shown in Figure 1 so that the whole multi-domain optical network appears to the service/network orchestrator as one optical network. The coordination of Packet/Optical multi-layer and IP/MPLS multi-domain is done by the
service/network orchestrator where it interfaces two IP/MPLS domain controllers and one optical L-MDSC.

Figure 5 shows how the Optical Domain Controllers create a new optical tunnel and the related interaction with IP/MPLS domain controllers and the NEs to bind the optical tunnel with proper forwarding instruction so that the VPN requiring hard isolation can be fulfilled.

As discussed in 2.2., in case that VPN has requirement for hard-isolated tunnel establishment, the service/network orchestrator will coordinate across IP/MPLS domain controllers and the NEs to establish the proper forwarding instruction for the optical tunnel.
controllers and Optical L-MDSC to ensure the creation of a new optical tunnel for the VPN in proper sequence. Figure 5 shows this scenario.

- The MDSC of the service/network orchestrator requests the L-MDSC to setup an Optical tunnel providing connectivity between the inter-layer interfaces at the ingress and egress PEs and requests the two IP/MPLS domain controllers to setup an inter-domain IP link between these interfaces.

- The MDSC of the service/network orchestrator then should provide the ingress IP/MPLS domain controller with the routing instruction for the VPN so that the ingress IP/MPLS domain controller would help its ingress PE to populate forwarding table. The packet with the VPN label should be forwarded to the optical interface the MDSC provided.

The Ingress Optical Domain PE needs to recognize MPLS-TE label on its ingress interface from IP/MPLS domain PE and encapsulate the MPLS packets of this MPLS-TE Tunnel into the ODU.

[Editor’s Note: We assumed that the Optical PE is LSR.]

The Egress Optical Domain PE needs to POP the ODU label before sending the packet (with MPLS-TE label kept intact at the top level) to the Egress PE in the IP/MPLS Domain to which the packet is destined.

[Editor’s Note: If there are two VPNs having the same destination CE requiring non-shared optical tunnels from each other, we need to explain this case with a need for additional Label to differentiate the VPNs]

2.5. Orchestrator-Controllers-NEs Communication Protocol Flows

This section provides generic communication protocol flows across orchestrator, controllers and NEs in order to facilitate the POI scenarios discussed in Section 2.3.2 for dynamic optical Tunnel establishment. Figure 6 shows the communication flows.
When Domain Packet Controller 1 sends the forwarding mapping information as indicated in (1) in Figure 6, the Ingress PE in Domain 1 will need to provision the VRF forwarding table based on the information it receives. Please see the detailed procedure in Section 2.3.1.2. A similar procedure is to be done at the Egress PE in Domain 2.

3. POI with VN Recursion Under Multiple Network Operators Control

[RFC8453] briefly introduces a case for the VN supplied to a customer may be built using resources from different technology layers operated by different operators. For example, one operator may run a packet TE network and use optical connectivity provided by another operator.
Figure 7 extracted from [RFC8453] shows the case where a customer asks for end-to-end connectivity between CE A and CE B, a virtual network. The customer’s CNC makes a request to Operator 1’s MDSC. The MDSC works out which network resources need to be configured and sends instructions to the appropriate PNCs. However, the link between Q and R is a virtual link supplied by Operator 2: Operator 1 is a customer of Operator 2.

To support this, Operator 1 has a CNC that communicates with Operator 2’s MDSC. Note that Operator 1’s CNC in Figure 10 is a functional component that does not dictate implementation: it may be embedded in a PNC.
The CMI in Figure 7 interfaces Operator 1’s CNC with Operator 2’s MDSC. The functions to perform and the information carried over the inter-operator CMI are identical to those of the Customer’s CNC and Operator 1’s MDSC. In other words, the two CMIs depicted in Figure 7 are recursive in nature.

3.1. Service Request Process between Multiple Operators

As discussed previously, the reclusiveness principle applies seamlessly over the two CMIs. This implies that Operator 1’s MDSC needs to pass all customer service requirements transparently to Operator 2’s MDSC so that Operator 2 should provision its underlay network tunnels to meet the service requirements of the original customer. The MDSC of Operator 1 should translate/map the original customer’s intent and service requirements and pass down to the corresponding PNC(s) which is(are) responsible for interfacing another operator (in this example, Operator 2) that provides transport services for the segment of the customer’s VN. The PNC in turn performs as a CNC when interfacing its southbound with Operator 2’s MDSC.

It is possible that additional recursive relationships may also exist between Operator 2 and other operators.

3.2. Service/network Orchestration of Operator 2
Operator 2 that provides transport service for Operator 1 may also need to perform service/network orchestration function just as the case for Operator 1.

4. Security Considerations

From a security and reliability perspective, ACTN may encounter many risks such as malicious attack and rogue elements attempting to connect to various ACTN components. Furthermore, some ACTN components represent a single point of failure and threat vector and must also manage policy conflicts and eavesdropping of communication between different ACTN components.

All protocols used to realize the ACTN framework should have rich security features, and customer, application and network data should be stored in encrypted data stores. Additional security risks may still exist. Therefore, discussion and applicability of specific security functions and protocols will be better described in documents that are use case and environment specific.

The CMI will likely be an external protocol interface. Suitable authentication and authorization of each CNC connecting to the MDSC will be required; especially, as these are likely to be implemented by different organizations and on separate functional nodes. Use of the AAA-based mechanisms would also provide role-based authorization methods so that only authorized CNC’s may access the different functions of the MDSC.

Where the MDSC must interact with multiple (distributed) PNCs, a PKI-based mechanism is suggested, such as building a TLS or HTTPS connection between the MDSC and PNCs, to ensure trust between the physical network layer control components and the MDSC. Trust anchors for the PKI can be configured to use a smaller (and potentially non-intersecting) set of trusted Certificate Authorities (CAs) than in the Web PKI. Which MDSC the PNC exports topology information to, and the level of detail (full or abstracted), should also be authenticated, and specific access restrictions and topology views should be configurable and/or policy based.
5. IANA Considerations

This document has no IANA actions.

6. Acknowledgements

The authors thank Adrian Farrel for useful discussions and their suggestions for this work.

7. References

7.1. Normative References


7.2. Informative References


8. Contributors

Authors’ Addresses

Y. Lee
Futurewei Technologies
Email: younglee.tx@gmail.com

D. Ceccarelli
Ericsson
Email: daniele.ceccarelli@ericsson.com

J. Tantsura
Apstra
Email: jefftant.ietf@gmail.com
Instant Congestion Assessment Network (iCAN) for Data Plane Traffic Engineering
draft-liu-ican-00

Abstract

iCAN (instant Congestion Assessment Network) is a set of mechanisms running directly on network nodes:

- To adjust the flows paths based on real-time measurement of the candidate paths.

- The measurement is to reflect the congestion situation of each path, so that the ingress nodes could decide which flows need to be switched from a path to another.

This is something that current SDN and TE technologies can hardly achieve:

- SDN Controller is slow and far from the data plane, it is neither able to assess the real-time congestion situation of each path, nor able to assure the data plane always go as expected (especially in SRv6 scenarios). However, iCAN can work with SDN perfectly: controller planning multi-path transmission, and iCAN does the flow optimization automatically.

- Traditional TE is not able to adjust the flow paths in real-time.

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

Traditional IP routing is shortest path based on static metrics, which can fulfill basic requirement of connectivity. MPLS-TE brings the capability of utilizing non-shortest paths, thus traffic dispatch...
is doable; however, MPLS-TE is only a complementary mechanism because of the scalability issue. Segment routing provides even more flexibility that paths could be easily programmed; and along with the controller, it could be scaled.

However, the above mentioned mechanism all run in the control plane, which implies that they are not able to sense the data plane situation in real-time, thus they are mostly for relative static planning/controlling (minuets, hours or even day-level) of network traffic and not able to adapt to the microscopic traffic change in real-time (e.g. mili-second level). So, in real bearer networks (metro, backbones etc.), it is always underload so that the redundant resources could tolerant the traffic burst, results in a significant waste of network resources.

This draft proposes the iCAN (Instant Congestion Assessment Network) architecture to achieve autonomous adapt to traffic changes in real-time in terms of switching flows between multiple forwarding paths. iCAN includes following things:

- A mechanism between ingress and egress nodes to assess the path congestion situation in RTT level speed, to recognize which paths are underload and which are heavy loaded.
- Recognizing big flows and small flows in the device, in real time
- Ingress node dispatches flows to multiple paths, to make load balance, or to guarantee SLA for specific flows

This draft also discusses use cases and implementation scenarios of iCAN.

2. iCAN Architecture and Key Technical Requirements

2.1. Architecture
As above figure shows, there are 3 entities:

1. Controller
   - Responsible for planning multiple paths for a set of flows that could be aggregated to a pair of Ingress/Egress routers.
   - After delivering the planned paths to the ingress router, the controller would need nothing to do.

2. Ingress router:
   - Serves as a local "controller" for the iCAN system.
   - Responsible for triggering the path congestion assessment, which is coordinated with the egress router through a measurement protocol.
   - After getting the assessment results, the ingress router would calculate which flows need to be switched to a different path, in order to make the paths load balanced or to assure the transport quality of a certain of important flows.
   - In order to do the path switching calculation, the ingress router needs to recognize the TopN flow passing by it, since switching the big flows would make the most effort.
3. Egress router:
   - Only needs to coordinate with the ingress router to do the path assessment.

2.2. Key technical requirements

2.2.1. Path quality assessment

   o Req-1: the assessment MUST reflex the congestion status of the paths. (Note: a candidate congestion metric is proposed as: [I-D.dang-ippm-congestion].)

   o Req-2: the assessment SHOULD be done within a RTT timeslot. Since iCAN is to adapt the traffic change in real-time, the assessment needs to be done very fast.

   o Req-3: the assessment MUST be done for multiple paths between the same ingress/egress routes simultaneously. (Note: a candidate congestion metric is proposed as: [I-D.dang-ippm-multiple-path-measurement].)

2.2.2. Recognition and statistic of flows in devices

   o Req-1: the device SHOULD be able to recognize TopN big flows within a timeslot.

   o Req-2: the device MAY need to statistic all flows’ amount within a timeslot.

2.2.3. Flow switching between paths

   o Req-1: the device SHOULD be able to recognize flow let. The flow switching is done from the next flow let.

   o Req-2: the device MAY need to actively generate gap to artificially create flow let. If the flow needs to be switched immediately, then the device would need to make the gap, to avoid out-of-order packets arriving to the destination through multiple paths.

   o Req-3: the device SHOULD avoid oscillation of frequently switching flows from one to another.

   o Req-4: multiple ingress devices SHOULD be able to coordinate so that they won’t switch flows to the shared path at the same time, to avoid potential congestion in the shared path.
3. Use Cases of iCAN

3.1. Network load balancing

Background problem: traffic is not balanced in current metro network.

While some links are heavily loaded, others might be still lightly loaded: unbalance could lows down the service quality (e.g. SLA could not be guaranteed in the heavily loaded links/tasks); unbalance could lows down the network utilization ratio (normally with 30%, e.g. a 100G physical capacity network can only bear at most 30G traffic, a huge waste of network infrastructure).

iCAN could be used for load balance among the multiple paths between a pair of ingress/egress nodes. Once the network is balanced, the real throughput of the network could be elevated significantly.

3.2. SLA assurance

Since iCAN could switch flow in real-time, it can guarantee a set of important flows. Once the path which carries the important flows is to be congested, the other flows could be switched to alternative paths, and the important flows would stably running in the original path.

(More content TBD)

3.3. Fine-Granularity reliability

Traditional reliability protocols such as BFD, can only assess the link on or off. With the path congestion assessment ability, iCAN could also asses the quality degradation.

(More content TBD)

4. Implementation Scenarios

4.1. iCAN with SRv6

- SR Multiple Explicit Paths

For example, there are 3 paths between the ingress and egress nodes, and the multi-path is defined as a SR-List containing LSP1/2/3.

The probe message detects the congestion status of the three SR-list paths. The edge device adjusts the load balancing between the three paths according to the congestion status of the three
SR-lists, and switch the flows from the path with a high congestion to the path with a low congestion.

- SR Multiple Explicit+Loose Paths

In loose path scenario, there needs to be an additional approach to probe the specific paths of a SR tunnel. After that, operations on the probed paths are the same as explicit path scenario.

4.2. iCAN with VxLAN

TBD.

4.3. iCAN with MPLS/MPLS-TE

TBD.

5. Standardization Requirements

1. Multi-path Planning (North Interface between Controller and devices)

2. Path Congestion Assessment (Horizontal Interface between devices), mostly regarding to Req-1&2&3 described in Section 2.2.1.

3. Flow Switching Negotiation (Horizontal Interface between devices), mostly regarding to Req-3&4 described in Section 2.2.3.

(More content TBD.)

6. Security Considerations

TBD.

7. IANA Considerations

TBD.

8. Acknowledgements

Very valuable comments were from Shunsuke Homma, Mikael Abrahamsson and Bruno Decraene.

A commercial router hardware based prototype had been implemented to prove the mechanisms discussed in the document are workable.
9. References

9.1. Normative References


9.2. Informative References


Author’s Address

Bing Liu
Huawei Technologies
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing 100095
P.R. China

Email: leo.liubing@huawei.com
Packet Network Slicing using Segment Routing
draft-peng-lsr-network-slicing-00

Abstract

This document presents a mechanism aimed at providing a solution for network slicing in the transport network for 5G services. The proposed mechanism uses a unified administrative instance identifier to distinguish different virtual network resources for both intra-domain and inter-domain network slicing scenarios. Combined with the segment routing technology, the mechanism could be used for both best-effort and traffic engineered services for tenants.

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 29, 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 to this document. Code Components extracted from this document must
According to 5G context, network slicing is the collection of a set of technologies to create specialized, dedicated logical networks as a service (NaaS) in support of network service differentiation and meeting the diversified requirements from vertical industries. Through the flexible and customized design of functions, isolation mechanisms, and operation and management (O&M) tools, network slicing is capable of providing dedicated virtual networks over a shared infrastructure. A Network slice instance (NSI) is the realization of network slicing concept. It is an E2E logical network, which comprises of a group of network functions, resources, and connection relationships. An NSI typically covers multiple technical domains, which includes a terminal, access network (AN), transport network (TN) and a core network (CN), as well as DC domain that hosts third-party applications from vertical industries. Different NSIs may have different network functions and resources. They may also share some of the network functions and resources.

For a packet network, network slicing requires the underlying network to support partitioning of 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 virtual networks. [I-D.ietf-teas-enhanced-vpn] described a framework to create virtual networks in a packet network. This document specifies a detailed mechanism to signal association of shared resources required to create and manage an NSI.

Currently there are multiple methods that could be used to identify the virtual network resource, such as Administrative Group (AG) described in [RFC3630], [RFC5329] and [RFC5305], Extended Administrative Groups (EAGs) described in [RFC7308], and Multi-Topology Routing (MTR) described in [RFC5120], [RFC4915] and [RFC5340]. However, all these methods are not sufficient to meet the requirements of network slicing service. For example, AG or EAG are limited to serve as a link color scheme used in TE path computation to meet the requirements of TE service for a tenant. But it is difficult to use them for an NSI allocation mapping (assuming that each bit position of AG/EAG represents an NSI) and, at the same time, as an IGP FIB identifier for best-effort service for the same tenant. MTR is limited to serve as an IGP logical topology scheme only used in the intra-domain scenario, and it is challenging to select inter-area link resource according to MT-ID when E2E inter-domain TE path needs to be created for a tenant.

Thus, there needs to be a new characteristic of NSI to isolate underlay resources. Firstly it could serve as TE criteria for TE service, and secondly, as an IGP FIB table identifier for best-effort service. This document introduces a new property of NSI called "Administrative Instance Identifier" (AII) and corresponding method of using it.

2. Conventions used in this document

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.

3. Overview of Mechanism

At the initial stage, each link in a physical network can be colored to conform with network slicing requirements. As previously mentioned, AII can be used to color links to partition underlay resource. Also, we may continue to use AG or EAG to color links for traditional TE purpose within a virtual network specified by an AII. A single or multiple AIIIs could be configured on each intra-domain or inter-domain link regardless of IGP instance configuration. At the minimum, a link always belongs to default AII (the value is 0). The number of AIIIs configured on a node’s links determines the number of
virtual networks the node is part of. This document defines a new extension of the existing IGP-TE mechanisms [RFC3630] and [RFC5305] to distribute AII information in an AS as a new TE parameter of a link. An SDN controller, using BGP-LS or another interface, will have a distinct view of each virtual network specified by AII.

Using the CSPF algorithm, a TE path for any best-effort (BE) or traffic engineered (TE) service can be calculated within a virtual network specified by the AII. The computation criteria could be <AII, min igp-metric> or <AII, traditional TE criteria> for the BE and TE respectively. Combined with segment routing, the TE path could be represented as:

- a single node-SID of the destination node, for the best-effort service in the domain;
- node-SIDs of the border node and the destination node, adjacency-SID of inter-domain link, for the inter-domain best-effort service;
- an adjacency-SID list, for P2P traffic engineered service.

Because packets of the best-effort service could be transported over an MP2P LSP without congestion control, SR best-effort FIB for each virtual network specified by AII to forward best-effort packets may be created in the IGP domain. Thus, CSPF computation with criteria <AII, min igp-metric> is distributed on each node in the IGP domain. That is similar to the behavior in [Flex-algo], but the distributed CSPF computation is triggered by AII.

To distinguish forwarding behavior of different virtual networks, prefix-SID need to be allocated per AII and advertised in the IGP domain.

For inter-domain case, in addition to the destination node-SID, several node-SIDs of the domain border node and adjacency-SID of inter-domain link are also needed to construct the E2E segment list. The segment list could be computed with the help of the SDN controller which needs to consider AII information during the computation. The head-end of the segment list maintains the corresponding SR-TE tunnel or [I-D.ietf-spring-segment-routing-policy].

As for the prefix-SID, adjacency-SID needs to be allocated per AII to distinguish the forwarding behavior of different virtual networks.

For P2P traffic engineering service, especially such as uRLLC service, it SHOULD not transfer over an MP2P LSP to avoid the risk of
traffic congestion. The segment list could consist of pure adjacency-SID per AII specific. The head-end of the segment list maintains the corresponding SR-TE tunnel or [I-D.ietf-spring-segment-routing-policy].

However, label stack depth of the segment list MAY be optimized at a later time based on local policies.

At this moment we can steer traffic of overlay service to the above SR best-effort FIB, SR-TE tunnel or SR policy instance, for the specific virtual network. The overlay service could specify a color for TE purpose, for example, color 1000 means <AII=10, min igp-metric> to say that I need best-effort forwarding within AII 10 resource, color 1001 means <AII=10, delay=10ms, AG=0x1> to say that I need traffic engineering forwarding within AII 10 resource, especially using link with AG equal to 0x1 to reach guarantee of not exceeding 10ms delay time. Service with color 1000 will be steered to an SR best-effort FIB entry, or an SR-TE tunnel/policy in case of inter-domain. Service with color 1001 will be steered to an SR-TE tunnel/policy.

4. Resource Allocation per AII

4.1. L3 Link Resource AII Configuration

In IGP domain, each numbered or unnumbered L3 link could be configured with AII information and synchronized among IGP neighbors. The IGP link-state database will contain L3 links with AII information to support TE path computation considering AII criteria. For a numbered L3 link, it could be represented as a tuple <local node-id, remote node-id, local ip-address, remote ip-address>, for unnumbered it could be <local node-id, remote node-id, local interface-id, remote interface-id>. Each L3 link could be configured to belong to a single AII or multiple AIIIs, for each <L3 link, AII> tuple it would allocate a different adjacency-SID. Note that an L3 link always belongs to default AII(0).

An L3 link that is not part of the IGP domain, such as the special purpose for a static route, or an inter-domain link, can also be configured with AII information and allocate adjacency-SID per AII as the same as IGP links. BGP-LS could be used to collect link state data with AII information to the controller.

4.2. L2 Link Resource AII Configuration

[I-D.ietf-isis-l2bundles] described how to encode adjacency-SID for each L2 member link of an L3 parent link. It is beneficial to deploy LAG or another virtual aggregation interface in network slicing.
scenario. Between two nodes, the dedicated link resources belong to
different virtual networks could be added or removed on demand, they
are treated as L2 member links of a single L3 virtual interface. It
is the single L3 virtual interface that needs to occupy IP resource
and be part of the IGP instance. Creating a new slice-specific link
on demand or removing the old one, is likely to affect some
configurations.

Each L2 member link of an L3 parent link SUGGESTED to be configured
to belong to a single AII, and different L2 member link will have
different single AII configuration, with different adjacency-SID. Note
that in this case, the L3 parent link belongs to default AII(0),
but each L2 member link belongs to the specific non-default AII.
Routing protocol control packets follow the L3 parent link of the L2
member link with the highest priority. At the same time, data
packets that belong to the specific virtual network will pass along
the L2 member link with the specific AII value.

TE path computation based on link-state database need view the
detailed L2 members of an L3 adjacency to select the expected L2 link
resource.

4.3. Node Resource AII Configuration

For topology resource, each node needs to allocate node-SID per AII
when it joins the related virtual network. All nodes in the IGP
domain can run CSPF algorithm with criteria <AII, min IGP metric> to
compute best-effort next-hop to any other destination nodes for a
virtual network AII-specific, based on the link-state database that
containing AII information so that SR best-effort FIB can be
constructed for each AII.

An intra-domain overlay best-effort service belongs to a virtual
network could directly match in the above SR best-effort FIB for the
specific AII, while an inter-domain overlay best-effort service
belongs to a virtual network could be over a segment list containing
domain border node-SID and destination node-SID which could match in
the above SR best-effort FIB for the specific AII.

5. Interworking with SR Flex-algorithm

[I-D.ietf-lsr-flex-algo] introduced a mechanism to do label stack
depth optimization for an SR policy in IGP domain part. As the color
of SR policy defined a TE purpose, traditionally the headend or SDN
controller will compute an expected TE path to meet that purpose. It
is necessary to map a color (32 bits) to an FA-ID (8 bits) when SR
flex-algorithm enabled for an SR policy, besides that, it is
necessary to enable the FA-ID on each node that will join the same FA
plane manually. However, the FAD could copy the TE constraints contained in the color template. We need to consider the cost of losing the flexibility of color when executing the flex-algo optimization, and also consider the gap between P2P TE requirements and MP2P SR LSP capability, to reach the right balance when deciding which SR policy need optimization.

5.1. Best-effort Service AII-specific

As described above, for best-effort service we have already constructed SR best-effort FIB per AII, that is mostly like [Flex-algo]. Thus, it is not necessary to map to FA-ID again for a color template which has defined a best-effort behavior within the dedicated AII. Of course, if someone forced to remap it, there is no downside for the operation, the overlay best-effort service (with a color which defined specific AII, best-effort requirement, and mapping FA-ID) in IGP domain will try to recurse over <AII, prefix> or <FA-ID, prefix> FIB entry.

5.2. Traffic Engineering service AII-specific

An SR-TE tunnel/policy that served for traffic engineering service of a virtual network specified by an AII was generated and computed according to the relevant color template, which contained specific AII and some other traditional TE constraints. If we config mapping FA-ID under the color template, the SR-TE tunnel/policy instance could inherit forwarding information from corresponding SR Flex-Algo FIB entry.

6. Examples

In this section, we will further illustrate the point through some examples. All examples share the same figure below.
Suppose that each link belongs to separate virtual network, e.g.,
link Ax belongs to the virtual network colored by AII A, link Bx
belongs to the virtual network colored by AII B. link x1 has an IGP
metric smaller than link x2, but TE metric larger.

To simplify the use case, each AS just contained a single IGP area.

6.1. intra-domain network slicing

From the perspective of node PE1 in AS1, it will calculate best-
effort forwarding entry for each AII instance (including default AII)
to destinations in the same IGP area. For example:

For <AII=0, destination=ASBR1> entry, forwarding information could be
ECMP during link A1 and link B1, with destination node-SID 100 for
<AII=0, destination=ASBR1>.

For <AII=A, destination=ASBR1> entry, forwarding information could be
link A1, with destination node-SID 200 for <AII=A,
destination=ASBR1>.

For <AII=B, destination=ASBR1> entry, forwarding information could be
link B1, with destination node-SID 300 for <AII=B,
destination=ASBR1>.

It could also initiate an SR-TE instance (SR tunnel or SR policy)
with the particular color template on PE1, PE1 is headend and ASBR1
is destination node. For example:

For SR-TE instance 1 with color template which defined criteria
including (default AII, min TE metric), forwarding information could
be ECMP during two segment list (adjacency-SID 1002 for <AII=0, link
A2>@PE1) and (adjacency-SID 1004 for <AII=0, link B2>@PE1).
For SR-TE instance 2 with the color template which defined criteria including \( \text{AII=A, min TE metric} \), forwarding information could be presented as the segment list \{ adjacency-SID 2002 for <AII=A, link A2>@PE1 \}.

For SR-TE instance 3 with the color template which defined criteria including \( \text{AII=B, min TE metric} \), forwarding information could be presented as the segment list \{ adjacency-SID 3004 for <AII=B, link B2>@PE1 \}.

Furthermore, we can use SR Flex-algo to optimize the above SR-TE instance. For example, for SR-TE instance 1, we can define FA-ID 201 with FAD that contains the same information as the color template, in turn, FA-ID 202 for SR-TE instance 2, FA-ID 203 for SR-TE instance 3. Note that each FA-ID also needs to be enabled on ASBR1. So that the corresponding SR FA entry could be:

For \(<\text{FA-ID=201, destination=ASBR1}>\) entry, forwarding information could be ECMP during link A2 and link B2, with destination node-SID 600 for \(<\text{FA-ID=201, destination=ASBR1}>\).

For \(<\text{FA-ID=202, destination=ASBR1}>\) entry, forwarding information could be link A2, with destination node-SID 700 for \(<\text{FA-ID=202, destination=ASBR1}>\).

For \(<\text{FA-ID=203, destination=ASBR1}>\) entry, forwarding information could be link B2, with destination node-SID 800 for \(<\text{FA-ID=203, destination=ASBR1}>\).

### 6.2. inter-domain network slicing via BGP-LS

An E2E inner-AS SR-TE instance with particular color template could be initiated on PE1, PE1 is head-end and PE2 is destination node. BGP-LS could be used to inform the SDN controller about the underlay network topology information including AII attribute. Thus the controller could calculate E2E TE path within the particular virtual network. For best-effort service, for example:

For SR-TE instance 4 with color template which defined criteria including \( \text{default AII, min IGP metric} \), forwarding information could be segment list \{ node-SID 100 for <AII=0, destination=ASBR1>, adjacency-SID 1001 for <AII=0, link A1>@ASBR1, node-SID 400 for <AII=0, destination=PE2> \}.

For SR-TE instance 5 with color template which defined criteria including \( \text{AII=A, min IGP metric} \), forwarding information could be segment list \{ node-SID 200 for <AII=A, destination=ASBR1>, adjacency-
SID 1001 for \(<\text{AII}=\text{A}, \text{link A1}@\text{ASBR1}, \text{node-SID 500 for } \langle\text{AII}=\text{A}, \text{destination=PE2}\rangle\).

For SR-TE instance 6 with color template which defined criteria including \((\text{AII}=\text{B}, \text{min IGP metric})\), forwarding information could be segment list \((\text{node-SID 300 for } \langle\text{AII}=\text{B}, \text{destination=ASBR1}\rangle, \text{adjacency-SID 1003 for } \langle\text{AII}=\text{B}, \text{link B1}@\text{ASBR1}, \text{node-SID 600 for } \langle\text{AII}=\text{B}, \text{destination=PE2}\rangle\}).

For TE service, for example:

For SR-TE instance 7 with color template which defined criteria including \((\text{default AII, min TE metric})\), forwarding information could be ECMP during two segment list \((\text{adjacency-SID 1002 for } \langle\text{AII}=0, \text{link A2}@\text{PE1}, \text{adjacency-SID 1001 for } \langle\text{AII}=0, \text{link A1}@\text{ASBR1}, \text{adjacency-SID 1002 for } \langle\text{AII}=0, \text{link A2}@\text{ASBR2}\rangle\) and \((\text{adjacency-SID 1004 for } \langle\text{AII}=0, \text{link B2}@\text{ASBR1}, \text{adjacency-SID 1003 for } \langle\text{AII}=0, \text{link B1}@\text{ASBR1}, \text{adjacency-SID 1004 for } \langle\text{AII}=0, \text{link B2}@\text{ASBR2}\rangle\}).

For SR-TE instance 8 with color template which defined criteria including \((\text{AII}=\text{A, min TE metric})\), forwarding information could be segment list \((\text{adjacency-SID 2002 for } \langle\text{AII}=\text{A, link A2}@\text{PE1}, \text{adjacency-SID 2001 for } \langle\text{AII}=\text{A, link A1}@\text{ASBR1}, \text{adjacency-SID 2002 for } \langle\text{AII}=\text{A, link A2}@\text{ASBR2}\rangle\}).

For TE service, if we use SR Flex-algo to do optimization, the above forwarding information of each TE instance could inherit the corresponding SR FA entry, it would look like this:

For SR-TE instance 7, forwarding information could be ECMP during two segment list \((\text{node-SID 600 for } \langle\text{FA-ID=201, destination=ASBR1}, \text{adjacency-SID 1001 for } \langle\text{AII}=0, \text{link A1}@\text{ASBR1}, \text{node-SID 600 for } \langle\text{FA-ID=201, destination=PE2}\rangle\rangle\) and \((\text{adjacency-SID 1004 for } \langle\text{AII}=0, \text{link B2}@\text{PE1}, \text{adjacency-SID 1003 for } \langle\text{AII}=0, \text{link B1}@\text{ASBR1}, \text{adjacency-SID 1004 for } \langle\text{AII}=0, \text{link B2}@\text{ASBR2}\rangle\}).

For SR-TE instance 8 with color template which defined criteria including \((\text{AII}=\text{A, min TE metric})\), forwarding information could be segment list \((\text{node-SID 700 for } \langle\text{FA-ID=202, destination=ASBR1}\rangle, \text{adjacency-SID 2001 for } \langle\text{AII}=\text{A, link A1}@\text{ASBR1}, \text{node-SID 700 for } \langle\text{FA-ID=202, destination=PE2}\rangle\}).
For SR-TE instance 9 with color template which defined criteria including \(\text{AII}=B\), \(\text{min TE metric}\), forwarding information could be segment list \{node-SID 800 for <FA-ID=203, destination=ASBR1>, adjacency-SID 3003 for <AII=B, link B1>@ASBR1, node-SID 800 for <FA-ID=203, destination=PE2>\).

6.3.  inter-domain network slicing via BGP-LU

In some deployments, operators adopt BGP-LU to build inter-domain MPLS LSP, overlay service will be directly over BGP-LU LSP. If overlay service has TE requirements that defined by a color, that means that BGP-LU LSP needs to have a sense of color too, i.e., BGP-LU label could be allocated per color. BGP-LU LSP generated for specific color will be over intra-domain SR-TE or SR Best-effort path generated for that color again.

In figure 1, PE2 can allocate and advertise six labels for its loopback plus color 1, 2, 3, 4, 5, 6 respectively. Suppose color 1 defines \(\text{default AII, min IGP metric}\), color 2 defines \(\text{AII}=A\), \(\text{min IGP metric}\), color 3 defines \(\text{AII}=B\), \(\text{min IGP metric}\), and color 4 defines \(\text{default AII, min TE metric}\), color 5 defines \(\text{AII}=A\), \(\text{min TE metric}\), color 6 defines \(\text{AII}=B\), \(\text{min TE metric}\). PE2 will advertise these labels to ASBR2 and ASBR2 then continues to allocate six labels each for prefix PE2 plus different color. Other nodes will have the same operation. Ultimately PE1 will maintain six BGP-LU LSP.

For example, the BGP-LU LSP for color 1 will be over SR best-effort FIB entry node-SID 100 for <AII=0, destination=ASBR1> to pass through AS1, over adjacency-SID 1001 for <AII=0, link A1>@ASBR1 to pass inter-AS, over SR best-effort FIB entry node-SID 400 for <AII=0, destination=PE2> to pass through AS2.

For example, The BGP-LU LSP for color 4 will over SR-TE instance 1 (see section 6.1), or SR best-effort FIB entry node-SID 600 for <FA-ID=201, destination=ASBR1> (see section 6.1) to pass through 6AS1, over adjacency-SID 1001 for <AII=0, link A1>@ASBR1 to pass inter-AS, over SR-TE instance 1’ or corresponding SR FA entry to pass through AS2.

7. Implementation suggestions

As a node often contains control plane and forwarding plane, a suggestion is that only default AII specific FTN entry need be installed on forwarding plane, so that there are not any modification and upgrade requirement for hardware and existing MPLS forwarding mechanism. FTN entry for non-default AII instance will only be maintained on the control plane and be used for overlay service iteration according to next-hop plus color (color will give AII
information and mapping FA-ID information). Note ILM entry for all AII need be installed on forwarding plane.

The same suggestion is also appropriate for SR Flex-algo.

8. IANA Considerations

TBD.

9. Security Considerations

TBD.

10. Acknowledgements

TBD.

11. Normative references

[I-D.ietf-isis-l2bundles]

[I-D.ietf-lsr-flex-algo]

[I-D.ietf-spring-segment-routing-policy]

[I-D.ietf-teas-enhanced-vpn]


Authors’ Addresses

Shaofu Peng
ZTE Corporation

Email: peng.shaofu@zte.com.cn

Ran Chen
ZTE Corporation

Email: chen.ran@zte.com.cn
Gregory Mirsky
ZTE Corporation

Email: gregimirsky@gmail.com
5G Transport Slice Connectivity Interface  
draft-rokui-5g-transport-slice-00

Abstract

The 5G Network slicing is an approach to provide separate independent E2E logical network from user equipment (UE) to applications where each network slice has different SLA requirements. Each E2E network slice consists of multitude of RAN-slice, Core-slice and Transport-slices, each with its own controller. To provide automation, assurance and optimization of the network slices, an E2E network slice controller is needed which interacts with controller in RAN, Core and Transport slices. The interfaces between the E2E network slice controller and RAN and Core controllers are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices.

The aim of this document is to provide the clarification of this interface and to provide the information model of this interface for automation, monitoring and optimization of the transport slices.
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 January 3, 2020.

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 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 ................................................. 3
   1.1. Definition of Terms .................................... 3
2. High level architecture of 5G end-to-end network slicing .... 5
3. Logical flow cross layers for automation of an end-to-end network slices ................................ 8
4. Definition of Transport Slice ............................... 11
   4.1. Transport Slices in Distributed RAN .................. 11
   4.2. Transport Slices in Centralized RAN (C-RAN) .......... 12
   4.3. Transport Slices in cloud RAN ........................ 13
   4.4. Transport Slice as a set of networks .................. 14
5. Transport Slice Connectivity Interface (TSCI) ............... 16
   5.1. Relationship between TSCI and various IETF data models 17
   5.2. Realization (aka Implementation) of transport slices 18
6. Transport slice connectivity interface (TSCI) information
1. Introduction

Network Slicing is a mechanism which a network operator can use to allocate dedicated infrastructures and services to a customer (aka tenant) from shared operator’s network. In particular a 5G network slice is inherently an E2E concept and is composed of multiple logical independent networks are created in a common operator’s network from an user equipement to applications. In specific, the network slicing receives attention due to factors such as diversity of services and devices in 5G each with its own SLA requirements. Each E2E network slice consists of multitude of RAN-slice, Core-slice and Transport-slices each with its own controller.

To provide automation, assurance and optimization of the network slices, an E2E network slice orchestrator is needed which interacts with controller of RAN, Core and Transport slices. The interfaces between the E2E network slice controller and RAN and Core controllers are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices. The aim of this document is to provide the clarification of this interface and to provide the information model of this interface for automation, monitoring and optimization of the transport slices.

1.1. Definition of Terms

Please refer to [I-D.homma-slice-provision-models] as well.

Customer: Also known as Tenant. Any application, client network, or customer of a network provider, i.e. an NS tenant is a person or group that rents and occupies NSIs from network provider.
E2E Network Slice: A E2E network slice is a virtual network provided by a Slice Provider that has the functionality and performance to support a specific industry sector and/or service. This capability will be the subject of a commercial agreement between the Slice Provider and Slice Buyer, and although it may well support dynamic scale-in/out, the Slice capability will normally be long-lasting i.e. only change on commercial timescales, although this may become more dynamic over time.

In specific, E2E 5G network slices are separate independent logical network E2E from user equipment (UE) to applications in a common infrastructure where each logical network has dedicated SLA. It is an E2E concept which spans multiple network domains (e.g. radio, transport and core), and in some cases more than one administrative domain.

Network Slice Instance (NSI): Also known as Network slice profile (NS profile), NSI is an E2E instance of a network slice blueprint which is instantiated in service provider’s network for specific customer and specific service type. Note that customer and service type can be wildcard.

Transport Slice: It is also called Transport Sub-Slice. A set of connections between various network functions (VNF or PNF) with deterministic SLAs. They can be implemented (aka realized) with various technologies (e.g. IP, Optics, FN, Microwave) and various transport (e.g. RSVP, Segment routing, ODU, OCH etc).

RAN Slice: It is also called RAN Sub-Slice. The context and personality created on RAN network functions for each E2E network slice.

Core Slice: It is also called Core Sub-Slice. The context and personality created on Core network (CN) functions for each e2e network slice.

S-NSSAI: Single Network Slice Selection Assistance Information, defined by 3GPP and is the identification of a Network Slice Sub-Slice: The RAN, Transport or Core Slices are also called Sub-Slices or Subnet; i.e. RAN, Transport and Core Sub-slices or Subnets

Service Level Agreement (SLA): An agreement between a customer (aka tenant) and network provider that describes the quality with which features and functions are to be delivered. It may include information on target KPIs (such as min guaranteed throughput, max tolerable latency, max tolerable packet loss rate); the types of
service (such as the network service functions or billing) to be executed; the location, nature, and quantities of services (such as the amount and location of compute resources and the accelerators require)

**gNB:** gNB incorporates two major modules; Centralized Unit (CU) and Distribute Unit (DU)

**UE:** UE: User Equipment such as vehicle Infotainment, Cell phone, IoT sensor etc

2. **High level architecture of 5G end-to-end network slicing**

To demonstrate the concept of 5G E2E network slicing and the role of various controllers consider a typical 5G network shown in Figure 1 where a mobile network operator Y has two customers (tenants) C1 and Public Safety. The boundaries of administrative domain of the operator includes RAN, Transport, Core and Application domains. Each customer requests to have several logical independent E2E networks from UEs (e.g. vehicle infotainment, mobile phone, IoT meters etc.) to the application, each with distinct SLAs. In 5G, each of these independent networks called an E2E network slice, which consists of RAN, Transport and Core slices (or RAN, Transport and Core Sub-slices).

In Figure 1 there are four E2E network slices, NS1 to NS4, each with its own RAN, Core and Transport slices. To create RAN slice, the RAN network functions such as eNB and gNB should be programmed to have a context for each E2E network slice. This context means that the RAN network functions should allocate certain resources for each E2E network slice such as air interface, various schedulers, policies and profiles to guarantee the SLA requirement for that specific network slice. By the same token, the Core slice means to create the context for each E2E network slice on Core network functions. This means that the RAN and Core network functions are aware of multiple E2E network slices.

When both RAN and Core slices are created, they should be connected together using a set of connections to have an E2E network slice. These set of connections are called Transport Slices, i.e. the transport slices are a group of connections with specific SLAs. The following factors dictate the number of transport slices. The details on transport slices will be discussed in see Section 4:

- The RAN deployment (i.e. distributed RAN, Centralized RAN or Cloud RAN). For example, in Cloud RAN, the RAN network function (i.e. gNB) is decomposed into two network functions (called CU and DU) where one or both will be VNFs and there is a Midhaul network.
between them. In this case, there will be a transport slice in RAN to connect DUs to CU.

- The location of the mobile applications. If there is a network between the mobile applications and the 5G Core, another transport slice is needed to connect the 5G Core to Applications.

- Mobile network policy on how to allow creation of the E2E network slices and if the sharing of transport slices between multiple E2E network slices are desirable and allowed.

At the end when RAN, Core and Transport slices are created, they should be bind or associated together to form a working E2E network slice. Since the nature of an E2E network slice is dynamic and the life cycle of each network slice might be a few hours or few months, various controllers are needed to perform the life cycle of various E2E network slices in their respective domains. As shown in Figure 1, each RAN, Core and Transport slice needs a controller. Also an E2E network slice controller is needed to provide the coordination and control of network slices from an E2E perspective.

In summary, an E2E network slice will involve several domains, each with its own controller: 5G RAN domain, transport domain, and 5G core domain. These need to be coordinated in order to deliver an E2E service. Note that in this context a service is not necessarily an IP/MPLS service but rather customer (aka tenant) facing services such as CCTV service, eMBB service and Infotainment service etc.
End to End Network Slices

RAN ----> Transport ----> Core ---->
Slice     Slice     Slice

E2E Network Slice Controller

RAN Controller | Transport Controller | Core Controller

Legend:
----- NS1: E2E network slice 1 for customer C1, service type Infotainment
===== NS2: E2E network slice 2 for customer C1, service type Autonomous Driving
+++++ NS3: E2E network slice 3 for customer C1, service type HD Map (NS3)
--- NS4: E2E network slice 4 for customer Public Safety, service type Video Surveillance

Customers (aka Tenants): Public Safety and C1
MNO (Mobile Network Operator): Operator Y

Figure 1: High level architecture of 5G end-to-end network slicing
3. Logical flow cross layers for automation of an end-to-end network slices

Figure 2 provides the logical flow cross layers to achieve the automatic creation of each E2E network slices such as NS1 mentioned in Figure 1. Below are the logical flow among various controllers to achieve this. It is important to note that these steps are logical and in practice some of them can be combined. For example, step 3 can be combined with steps 4 or 5.

1. The customer C1 will request the Operator Y for creation of an E2E network slice for Infotainment service type and SLA of 10 Mbps

2. The E2E network slice controller receives this request and using its pre-defined network slice blueprints (aka network slice templates) creates a network slice profile (aka network slice instance) which contains all the network functions in RAN and core which should be part of this E2E network slice. It then goes through decomposition of this profile and triggers various actions. Steps 3 to 7 shows these actions.

3. A request for creation of the RAN slice will be triggered to RAN Controller.

4. If RAN network functions are virtual, the RAN Slice controller triggers the creation of VNFs in RAN (using for example ETSI interface Os-Ma-nfvo)

5. NFVO manages the life cycle of virtual RAN network functions

6. Since both physical and virtual RAN network functions which are part of the E2E network functions are known to RAN controller, it then triggers the creation of RAN slice by programming RAN network functions

7. Similar to previous steps, a request for creation of the Core slice will be triggered to Core Controller.

8. If Core network functions are virtual, the Core Slice controller triggers the creation of VNFs (using for example ETSI interface Os-Ma-nfvo) and NFVO manages the life cycle of virtual Core network functions

9. Since both physical and virtual Core network functions which are part of the E2E network functions are known to Core controller, it then triggers the creation of Core slice by programming Core network functions
10. In this step, various transport slices (i.e. various connections) need to be created between various network functions, e.g. transport slices between RAN and Core slices, transport slices between RAN network functions or transport slices between core and applications.

11. The transport slices will be implemented (aka realized) in the network.

12. [Optional] If the creation of transport slice involves creation of VNFs (e.g. Firewall, security gateway etc.), triggers the creation of these VNFs (using for example ETSI interface Os-Ma-nfvo).

13. The E2E network slice controller binds (or associates) all these slices together to form a single E2E network slice for specific customer and specific service type.

14. At the end, when the E2E network slice is created, the E2E network slice controller will allocate a unique network slice id (called S-NSSAI) and eventually during the UE network attach, all the UEs will be informed about the existence of this newly created E2E network slice and then they can request it using the 3GPP 5G signaling procedures.

Note that the interfaces 3 and 7 between the E2E network slice Controller and RAN and Core controllers and their information models are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices, i.e. interface 10. The aim of this document is to define this interface. More details will be provided in Section 5.
Figure 2: Logical flow cross layers for automation of an end-to-end network slices
4. Definition of Transport Slice

Since the transport slice is an important concept throughout this document, this section describes this concept in more detail. To this end, consider Figure 3 where a group of physical or virtual network functions (PNF, VNF or both) are connected together. In particular, a single transport slice is defined as:

- A distinct set of connections between multiple VNFs and PNFs each with its own deterministic SLA which is implemented (aka realized) in the network using any technology (e.g. IP, Optics, Microwave and PON), any tunnel types (e.g. IP, MPLS, SR, ODU/OCH) and any L0/L1/L2/L3 service types.

```
<table>
<thead>
<tr>
<th>NF1n</th>
<th></th>
<th>NF21</th>
</tr>
</thead>
<tbody>
<tr>
<td>NF11</td>
<td>(                      )</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(Transport Slice)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(                      )</td>
<td>-----</td>
</tr>
<tr>
<td>NF12</td>
<td>(A set of distinct)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(connetions connecting)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(physical or virtul)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(network functions)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(NF11, NF12, ..., NF1n to)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(NF21, ..., NF2m)</td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td>(                      )</td>
<td>-----</td>
</tr>
</tbody>
</table>
```

Figure 3: Definition of transport slice

For clarity, in next three sections, a few examples of the transport slices will be provided for following RAN deployments:

- Distributed RAN
- Centralized RAN (C-RAN)
- Cloud RAN

4.1. Transport Slices in Distributed RAN

Distributed RAN is the most common deployment of 4G and 5G RAN networks where as shown in Figure 4, the RAN network is connected to Core network using the transport network. Optionally the mobile applications can be also connected to the Core network using another
overlay network (e.g. Internet where the mobile applications are virtualized in another data center).

In this case, for a single E2E network slice, in addition to RAN and Core slices, there are two sets of transport slices; TS_1 and TS_2. TS_1 is connecting the RAN to Core and TS_2 is connecting Core to Applications.

\[ \text{---------- End to End Network Slice ----------} \]
\[ \text{<--- RS --><--- CS --->} \]
\[ \text{<--- TS_1 --->} \]
\[ \text{<--- TS_2 --->} \]

Legend
TS: Transport Slice
RS: RAN Slice
CS: Core Slice

Figure 4: Transport slices in distributed RAN

4.2. Transport Slices in Centralized RAN (C-RAN)

The RAN consists of two functional units, the baseband unit (BBU) and the radio unit (RU). The BBU processes the signal and is connected to the transport network. The RU transmits and receives the carrier signal that is transmitted over the air to the end user equipment (UE). In Centralized RAN (aka C-RAN) as depicted in Figure 5, the RU and BBU are separated by a network called fronthaul network. In this case a single E2E network slice contains three set of transport slices TS_1, TS_2 and TS_3 where TS_1 and TS_2 are identical to distributed RAN case (see Figure 4) and the new TS_3 connects the Radio Units (RU) to the BBUs.
4.3. Transport Slices in cloud RAN

In new cloud-RAN architecture, the baseband unit functionality is further divided into real-time and non-real-time. The former is deployed close to antenna to manages the real-time air interface resources while the non-real-time control functions are hosted centrally in the cloud. In 5G, BBU is split into two parts called CU (Central Unit) and DU (Distributed Unit) as shown in Figure 6 where these entities are connected by new network called Midhaul.

In this deployment for a single E2E network slice, there are four sets of transport slices TS_1, TS_2, TS_3 and TS_4 where TS_1 and TS_2 and TS_3 are identical to distributed and centralized RAN (see Figure 4 and Figure 5). The new transport slice TS_4 will connect DUs to CUs.
To further explore the content of a transport slice, let's focus on transport slice TS_1 between RAN and Core. Note that the following discussion is also applicable to any other transport slices TS_2, TS_3, or TS_4. As shown in Figure 7 for an individual E2E network slice belongs to a specific customer for a specific service type, the RAN domain is connected to Core domain through the transport network. In this example, the E2E network slice is identified by n-nssai=10 for customer C1 and service type Infotainment. Two RAN network elements BBU1 and BBU2 and two Core network elements AMF1 and UPF1 are all part of the E2E network slice. There are two sets of distinct connections between RAN and Core domains;

- Network-C which connects BBU1 and BBU2 to AMF1 with SLA-C
- Network-U which connects BBU1 and BBU2 to UPF1 with SLA-U
Customer: C1
Service type: Infotainment
S-NSSAI: 10
Network-C (from BBU-1.tp1, BBU-2.tp1 to AMF1.tp1 with SLA-C)
Network-U (from BBU-1.tp2, BBU-2.tp2 to UPF1.tp2 with SLA-U)

Transport slice TS_1: { Network-C and Network-U }

Figure 7: Transport Slice as a set of networks

Note that the SLA-C and SLA-U can be different. The combination of these two networks is called a single transport slice TS_1. Note that the definition of the transport slice does not specify how these connections should be realized (or implemented). It also does not specify which technology (e.g. IP, MPLS, Optics etc.) or tunnel type (e.g. MPLS, Segment Routing etc.) should be used during the realization. Those are part of realization of the transport slice done by transport slice controller. With this approach the definition is logically separated from implementation of transport slices. This gives a tremendous programmability and flexibility during the realization of transport slices using any type of technologies and tunnel types.
In summary, a transport slice is a distinct set of technology-agnostic connections each with its own deterministic SLA which can be implemented (aka realized) using any technology, tunnel type and service type.

5. Transport Slice Connectivity Interface (TSCi)

As shown in Figure 2, the interfaces 3 and 7 between the E2E network slice Controller and RAN and Core controllers, respectively and their information models are defined in various 3GPP technical specifications [TS.28.530-3GPP], [TS.28.531-3GPP], [TS.28.540-3GPP] and [TS.28.541-3GPP]. However, 3GPP has not defined the same interface for transport slices, i.e. interface 10. The aim of this document is to provide the clarification of this interface and to provide the information model of this interface. The interface is called the Transport Slice Connectivity interface (TSCi) which provides some means for automation, monitoring and optimization of the transport slices.

This new interface is needed in order to simplify the creation of the Transport slices and hides all the complexity of implementation (aka realization) from higher E2E network slice controller similar to their RAN and Core counterparts defined by 3GPP.

The aim of this document is to define a new interface and its information model between the E2E network slice controller and the transport slice controller. The characteristics on this new interface are:

a. The interface allows a request and response about resources. It should not allow negotiation, as this would be complex and not have a clear benefit

b. This interface is needed by the same layer that configures 3GPP RAN and Core slices to support the E2E path management in 5G networks. The provider of this interface is the higher layer controller which needs to create a connectivity between two network functions. The provider of this interface is the lower layer controller which provide the implementation (aka realization) of this connectivity (i.e. transport slice).

c. This interface is needed in industry to achieve true standard based automation of 5G E2E network slices. It provides a technology-agnostic intent-based interface to the E2E network slice controller similar to interfaces defined by 3GPP for RAN and Core slices.
d. This interface is independent of type of network functions which needs to be connected, i.e. it is independent of any specific repository, software usage, protocol, or platform which realizes the physical or virtual network functions.

e. The interface standardizes a way to learn about what resources are available in the network which impact the creation of the transport slices.

f. This technology independent interface simplifies the creation of the transports slices by describing it in a standard way along with all the network functions (such as eNB, gNB, CU, DU, Core, Mobile application server, etc.) that compose a transport slice, their properties, attributes and their SLA requirements, i.e. the connectivity resources requested from an E2E network slice controller to transport slice controller with their corresponding SLA requirements.

g. This interface provides capabilities for transport slice monitoring and analytics. It means that the TSCi interface allows enabling/disabling the collection of the transport slices telemetry, assurance and Threshold Crossing Alert (TCA) data and providing them to the E2E network slice controller.

h. This interface provides capabilities for optimization of the transport slices. Since the nature of an E2E network slice is dynamic, it is important to make sure the transport slice SLA are valid and in case any SLA violation happens, the transport slice controller performs the closed-loop action and informe the upper layer E2E network slice controller for the violation and closed-loop action. This interface allows this functionality.

i. This interface allows binding and association between RAN to Transport slices and between Core to Transport slices.

j. This interface complements various IETF service, tunnel, path models by providing an abstract layer on top of these models.

5.1. Relationship between TSCi and various IETF data models

The transport slice connectivity interface and its relationship to various IETF data models are not addressed in any IETF WGs as it has very unique characteristics of the 5G E2E network slicing. The new interface complements various IETF service, tunnel, path data models by providing an abstract layer on top of these models.
Figure 8: Relationship between transport slice interface and IETF Service/Tunnels/Path data models

Figure 8 shows more details on how the new transport slice connectivity interface (TSCi) relates to various IETF service/tunnels/path models. The transport slice controller receives a request for creation of a transport slice. This request is an abstract intent-based request which contains the required connections between various network functions in RAN and Core. The transport slice controller will then realize or implement those connections using various IETF models.

In a sense, the new transport slice connectivity interface provides an abstract layer on top of IETF models. The IETF service, path and tunnel data models can be any existing IETF service models such as L3SM or L2SM ([RFC8049] and [RFC8466]). It also can be any future data models.

5.2. Realization (aka Implementation) of transport slices

Since the transport slice connectivity interface describes the connections not the services, it is essential to make a distinction between connections and implemented services. Referring to Figure 9, assume using the new transport slice interface, the E2E network slice controller requests the creation of a transport slice which has multiple connections between RAN and Core. One of these connections is shown below between RAN1 and UPF1. The E2E network slice controller can request a connection between RAN1 to UPF1 for a
specific tenant, specific service type and specific SLA (e.g. customer Public Safety for service CCTV with latency of 5 [ms] or better).

![Diagram of transport network](attachment:transport_network_diagram.png)

Legend:

- Connection part of the transport slice
- Implementation (aka realization) of the transport slice

Figure 9: Distinction between Connections and Services

To realize (aka implement) this connection, the transport slice controller, will find the endpoint for the L0/L1/L2/L3 services, find the best path and create a service between these endpoints. In this example, in order to implement the connection between RAN1 and UPF1, the transport slice controller will first find the best border routers BR1 and BR2, find the best path between them and finally creates a Service/path from BR1 to BR2. It is important to note that:

- The endpoints of the Connection are different from the endpoints of the Services, paths and tunnels. This is the unique characteristic of transport slices where the endpoints of the connections are different from endpoints of the Services (i.e. endpoint of the transport slices are RAN1 and UPF1 whereas the endpoint of services are BR1 and BR2)
o The Service/path API can be any existing IETF service models such as L3SM or L2SM ([RFC8049] and [RFC8466]). It also can be any future service model

o In order to have the connectivity between RAN1 and UPF1, the RAN and Core slices should be associated to Transport slice. This is also a by-product of the Transport slice connectivity interface when all allocated resources for access points (such as allocated VLAN IDs, IP addresses etc) are conveyed to RAN and Core Slices. This will be done by coordination between transport slice controller and E2E network slice controller.

6. Transport slice connectivity interface (TSCI) information model

   Based on the definition of a transport slice (see Section 4), the high-level information model of a transport slice connectivity interface should conform with Figure 10:
The proposed transport slice information model should include the following building blocks:

- transport-slice-info: Information related to transport slice
- network-slice-info: A list of all E2E network slices mapped to transport slice
- transport-slice-networks: Each transport slice is a set of networks. Each network contains:
  * list of nodes (i.e. list of RAN and Core network functions)
  * list of connection-links (i.e. list of connections between nodes)
* list of transport-slice-policies (i.e. various SLA, Selection and Monitoring policies)

6.1. transport-slice-info

It contains information such as transport slice name, transport slice ID etc. The details will be provided in next version of this draft.

6.2. network-slice-info

As discussed in Section 3, a request for creation of a transport slice starts with the fact that a customer (aka tenant) will request an E2E network slice from an operator for a certain service type (e.g. CCTV, Infotainment, URLLC, etc.). So, there is a mapping between transport slice and the E2E network slice. Depending on the deployment, it is possible to map multiple E2E network slice to a transport slice, i.e. the mapping between transport slice to E2E network slice is one to many.

The network-slice-info contains the list of E2E network slices which are mapped to the transport slice with all relevant information such as S-NSSAI, name of customer, service type etc. The details will be provided in next version of this draft.

6.3. transport-slice-networks

As per Section 4, a transport slice will contain one or more transport-slice-networks.

6.4. node

As discussed in Section 4, the transport slice comprises one or more connectivity networks between RAN and Core network elements. It is also possible to have the connectivity networks between RAN and RAN network elements for some RAN deployments (example for midhaul network). As discussed in section 2.2, when the transport slice is realized (implemented), the network elements which are the endpoints of realization of the transport slice might be different. As a result, the nodes in this model represent RAN or Core network elements. However, the model is flexible where nodes might represent the routers or switches which are the endpoints of the transport slice realization.

The attributes of the node are IP address, node name, domain ID and termination points. The details will be provided in next version of this draft.
6.5.  connection-link

Each transport-slice-networks contain one or more connections between various nodes described in Section 6.4. The connection-link is a list of links which are represented by endpoint-A, endpoint-B etc. The details will be provided in next version of this draft.

6.6.  transport-slice-policy

To establish a transport slice, one or more transport-slice-networks will be created each with certain SLA requirement which is represented by transport-slice-policy. This draft proposes three types of transport slice policies to be supported:

- transport-slice-sla-policy
- transport-slice-selection-policy
- transport-slice-assurance-policy

The summary of these policies will be provided here. The details will be provided in next version of this draft.

6.6.1.  transport-slice-sla-policy

This is a mandatory policy. The transport-slice-policy represents in a generic and technology-agnostic way the SLA requirement needed to realize a group of connections which are part of a transport slice. It is defined per transport-slice-network. It contains the bounded latency, bandwidth, reliability, security etc.

6.6.2.  transport-slice-selection-policy

This is an optional policy. In some deployment, the E2E network slice controller might want to assist the transport slice controller on how to realize a transport slice by providing some information regarding the type of technologies and tunnels. This information will be provided in transport-slice-selection-policy.

6.6.3.  transport-slice-assurance-policy

This is a mandatory policy. The E2E network slice controller shall influence the transport slice controller for transport slice assurance on how to perform monitoring, analytics and optimization. To this end, the transport-slice-assurance-policy will be used. It contains, the type of assurance needed, time interval, how often inform the E2E network slice controller etc.
6.7. IETF models

Currently none of the IETF data models address the modeling of transport slice connectivity as shown in Figure 6. However, the various IETF data models might be augmented to address the information model of the transport slice connectivity interface. Following is the list of candidates IETF YANG models. This is not an extensive list and the next version of the draft will provide a more comprehensive list.

6.7.1. ACTN model

As defined in [RFC8453], [I-D.king-teas-applicability-actn-slicing] and [I-D.ietf-teas-actn-vn-yang] a VN (Virtual Network) is the abstract customer view of the TE network. 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 E2E tunnel across the underlying networks.

This definition is very similar to definition of the transport slice which means that the VN YANG model can be augmented to address the modeling of the transport slice shown in Figure 6. This is work in progress for next version of this document.

6.7.2. i2rs model

Similar to [I-D.qiang-coms-netslicing-information-model], the data model for network topologies developed in [[I-D.ietf-i2rs-yang-network-topo] can be augmented to address the modeling of the transport slice. This is work in progress for next version of this document.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

TBD

9. Informative References

[I-D.boucadair-connectivity-provisioning-protocol]
[I-D.homma-slice-provision-models]
Homma, S., Nishihara, H., Miyasaka, T., Galis, A., OV, V.,
Lopez, D., Contreras, L., Ordonez-Lucena, J., Martinez-
Julia, P., Qiang, L., Rokui, R., Ciavaglia, L., and X.
Foy, "Network Slice Provision Models", draft-homma-slice-
provision-models-00 (work in progress), February 2019.

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

[I-D.ietf-teas-actn-vn-yang]
Lee, Y., Dhody, D., Ceccarelli, D., Bryskin, I., Yoon, B.,
Wu, Q., and P. Park, "A Yang Data Model for VN Operation",
draft-ietf-teas-actn-vn-yang-04 (work in progress),
February 2019.

[I-D.king-teas-applicability-actn-slicing]
King, D. and Y. Lee, "Applicability of Abstraction and
Control of Traffic Engineered Networks (ACTN) to Network
Slicing", draft-king-teas-applicability-actn-slicing-04
(work in progress), October 2018.

[I-D.qiang-coms-netslicing-information-model]
Qiang, L., Galis, A., Geng, L.,
kiran.makhijani@huawei.com, k., Martinez-Julia, P.,
Flinck, H., and X. Foy, "Technology Independent
Information Model for Network Slicing", draft-qiang-coms-
netslicing-information-model-02 (work in progress),
January 2018.

[RFC7297]  Boucadair, M., Jacquenet, C., and N. Wang, "IP
Connectivity Provisioning Profile (CPP)", RFC 7297,
DOI 10.17487/RFC7297, July 2014,

[RFC8049]  Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data
Model for L3VPN Service Delivery", RFC 8049,
DOI 10.17487/RFC8049, February 2017,
<https://www.rfc-editor.org/info/rfc8049>.

Abstraction and Control of TE Networks (ACTN)", RFC 8453,
DOI 10.17487/RFC8453, August 2018,


Authors' Addresses

Reza Rokui
Nokia
Canada

Email: reza.rokui@nokia.com
Abstract

This document describes the use of RSVP (Resource Reservation Protocol), including all the necessary extensions, to establish Point-to-Point (P2P) Traffic Engineered IP (IP-TE) Label Switched Path (LSP) tunnel(s) for use in native IP forwarding networks.

This document proposes specific extensions to the RSVP protocol to allow the establishment of explicitly routed IP paths using RSVP as the signaling protocol. The result is the instantiation of an IP Path which can be automatically routed away from network failures, congestion, and bottlenecks.

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 January 9, 2020.

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.
Table of Contents

1. Introduction .............................................. 2
2. Terminology ............................................. 3
  2.1. Acronyms ........................................... 3
3. Overview of IP LSP Tunnels ............................... 4
  3.1. Creation and Management .............................. 4
  3.2. Path Maintenance .................................. 5
  3.3. Signaling Extensions ................................. 5
       3.3.1. RSVP Path message ........................... 5
  3.4. RSVP Resv Label Object .............................. 6
  3.5. EAB address Handling ................................ 7
       3.5.1. Egress Router ................................ 7
       3.5.2. Ingress and Transit Router .................... 7
  3.6. Protection .......................................... 8
  3.7. Shared Forwarding .................................. 8
  3.8. Error Conditions ................................... 9
4. IANA Considerations ..................................... 9
5. Security Considerations ................................. 9
6. Acknowledgement ......................................... 9
7. Contributors ............................................ 9
8. References ............................................. 9
   8.1. Normative References ............................... 9
   8.2. Informative References ............................. 10
8. Authors’ Addresses ...................................... 11

1. Introduction

In native IP networks, each router runs a routing protocol to determine the best next-hop(s) to a specific destination. The best next-hop(s) are usually determined by favoring those that run along the shortest path to the destination. When data flows across the network, it is routed hop-by-hop and follows the selected path by each hop towards that destination on each hop.

It is sometimes desirable for an ingress router to be able to steer traffic towards a destination along a pre-determined or pre-computed path that may follow a path other than the default shortest path. For example, some flows may require to be forwarded along the least latency path. Others, may desire to be routed with bandwidth guarantees along the selected path, or along a path that honors
certain resource affinities or Shared Risk Link Group (SRLG) memberships.

A solution to such use-cases entails: 1) router(s) in the network to be able to maintain and disseminate per link state information, 2) ingress routers or an external server to be able to perform a stateful path computation for feasible path(s) on top of the network topology, and 3) for ingress router(s) to be able to steer or tunnel the traffic along the established path towards the destination.

Mechanisms have been defined to achieve this with RSVP extensions for Traffic Engineered Multiprotocol Label Switching (MPLS-TE) networks as described in [RFC3209]. This document proposes extensions to the existing mechanisms for achieving this in networks that rely on native IP for their forwarding.

This document covers the necessary extensions for establishing Point-to-Point (P2P) Traffic-Engineered IP (IP-TE) Label Switched Path (LSP) Tunnels. The equivalent extensions needed for setting up multicast IP-TE LSPs are currently out of the scope of this document.

2. 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.

2.1. Acronyms

The reader is assumed to be familiar with the terminology used in [RFC2205] and [RFC3209].

IP-TE LSP (Traffic Engineered IP Label Switched Path):

The path created by programming of an IP route along the explicitly specified or dynamically computed sequence of router hops, allowing an IP packet to be forwarded from one hop to another along the established path.

IP-TE LSP Tunnel:

An IP-TE LSP which is used to tunnel traffic over the pre-established IP path.

Traffic Engineered IP Tunnel (IP-TE Tunnel):
A set of one or more IP-TE LSP Tunnels which carries a traffic trunk.

3. Overview of IP LSP Tunnels

IP-TE LSP tunnels are established over a native IP forwarding network. In many cases, IP-TE LSP(s) are explicitly routed from an ingress router. The explicit route used to establish an IP-TE LSP may be locally computed at the ingress router, or externally computed by an entity such as a Path Computation Element (PCE) [RFC4655].

To support the setup of IP-TE LSP tunnel(s), the egress routers reserve one or more local IP prefixes or Egress Address Block(s) (EABs) that are dedicated for RSVP to establish IP-TE LSP(s) tunnels. The EAB(s) addresses at the egress router are only managed by the RSVP protocol and are not required to be exchanged by any other routing protocol.

It is possible in some cases, where the IP-TE LSP(s) are contained within a single administrative domain boundary, for EAB(s) to be allocated from the private IP address space as defined in [RFC1918] or from the unique-local space as defined in [RFC4193] and [RFC5156].

Also useful in some applications for sets of IP-TE LSP tunnels to be associated together to facilitate reroute operations or to spread a traffic trunk over multiple IP-TE LSP tunnel paths. For traffic engineering applications to IP-TE LSP tunnel(s), such sets are called traffic engineered tunnels (TE IP tunnels).

3.1. Creation and Management

An IP-TE LSP tunnel is unidirectional in nature. To create an IP-TE LSP tunnel, the ingress router of the IP-TE LSP tunnel creates an RSVP Path message with a session type of LSP_TUNNEL_IPv4 or LSP_TUNNEL_IPv6 and follows the procedures outlined in [RFC3473] to insert a Generalized Label Request object into the Path message. The Generalized Label Request object indicates that an IP address binding is requested to the IP-TE LSP tunnel. The binding of an EAB address to an IP-TE LSP tunnel happens at the egress router and is signaled using an RSVP Resv message sent from the egress router.

The ingress router uses a pre-computed explicit path to populate the EXPLICIT_ROUTE object that is added in the RSVP Path message. The explicitly routed path can be administratively specified, or automatically computed by a suitable entity based on QoS and policy requirements, taking into consideration the prevailing network state. In addition, RSVP-TE signaling [RFC3209] allows for the specification
of an explicit path as a sequence of strict and loose routes. Such combination of abstract nodes, and strict and loose routes significantly enhances the flexibility of path definitions.

The ingress MAY also add a RECORD_ROUTE object to the RSVP Path message in order to receive information about the actual route traversed by the IP-TE LSP tunnel. The RECORD_ROUTE object MAY also be used by the egress router to determine whether Shared Forwarding as described in Section 3.7 is possible amongst different IP-TE LSP tunnel(s).

3.2. Path Maintenance

If the ingress router discovers a better path, after an IP-TE LSP tunnel has been successfully established, it can dynamically reroute the session by changing the EXPLICIT_ROUTE object. If problems are encountered with the EXPLICIT_ROUTE object, either because it causes a routing loop or because some intermediate routers do not support it, the ingress is notified.

Make-before-break procedures can also be employed to modify the characteristics of an IP-TE LSP tunnel. As described in [RFC3209], the LSP ID in the Sender Template object is updated in the new RSVP Path message that is signaled. As usual, the combination of the LSP_TUNNEL SESSION object and the SE reservation style naturally accommodates smooth transitions in bandwidth and routing.

For example, to trigger a bandwidth increase, a new RSVP Path Message with a new LSP_ID can be used to attempt a larger bandwidth reservation while the current LSP_ID continues to be refreshed to ensure that the reservation is not lost if the larger reservation fails.

3.3. Signaling Extensions

This section describes RSVP signaling extensions and modifications to existing RSVP objects that are carried in RSVP Path or Resv messages and are required to establish IP-TE LSP tunnel(s).

3.3.1. RSVP Path message

To signal an IP-TE LSP tunnel, the Generalized Label Request object is carried in the RSVP Path message and used to request an IP address binding to the IP-TE LSP tunnel.

The Generalized Label Request is defined in [RFC3471] and has the below format:
To request an IPv4 or IPv6 binding to an IP-TE LSP tunnel, the Generalized Label object carries the following specifics:

1. the LSP encoding type is set to Packet (1) [RFC3471].
2. the LSP switching type is set to "IPv4-TE" (TBD1), or IPv6-TE (TBD2)
3. the Generalized Payload Identifier (G-PID) MAY be set to All (0) or in some cases to the specific payload type if known, e.g. Ethernet (33) [RFC3471].

3.4. RSVP Resv Label Object

The egress is responsible to bind an IP EAB address to an IP-TE LSP tunnel.

Once the egress router receives the RSVP Path message with the Generalized Label Request object containing the parameters described in Section 3.3.1, the egress router determines and binds an EAB address to the newly established IP-TE LSP tunnel. Note, subject to a local policy and additional path check(s), the egress MAY assign an already in used EAB address to the newly established IP-TE LSP tunnel.

The RSVP Resv message that is created by the egress router uses the Generalized Label defined in [RFC3471] to carry the EAB address that is bound to newly established IP-TE LSP tunnel.

The RSVP Generalized Label object has the following format:

LABEL class = 16, C_Type = 2

The information carried in a Generalized Label is:
Label (Variable Length):

Carries label information. The interpretation of this field depends on the parameters signaled in the Generalized Label Request.

3.5. EAB address Handling

The RSVP Resv message that is created by the egress router is forwarded upstream along the signaling path towards the ingress router. Each router starting from the egress will perform the following steps when binding the EAB address to the IP-TE LSP tunnel.

3.5.1. Egress Router

The egress router manages the EAB addresses for the use of establishing IP LSP tunnel(s).

The egress router MAY assign unique EAB address to newly established IP-TE LSP tunnel(s) and MAY free an existing EAB address upon destroying a previously established IP-TE LSP tunnel. Note that an egress router MAY hold on to an EAB when the IP-TE LSP is being destroyed if it determines other IP-TE LSP(s) are sharing it.

Once an EAB address is allocated and bound to a new IP-TE LSP tunnel, the egress router programs the address in its forwarding table as local address – hence, resulting in decapsulation of the outer IP header on any packet arriving over the IP-TE LSP tunnel and hence yielding the original IP datagram that was tunneled over the IP LSP tunnel.

3.5.2. Ingress and Transit Router

A transit or an ingress router extracts the EAB address that the egress router binds to the IP-TE LSP tunnel from the Generalized Label object contained in the RSVP Resv message that is propagated upstream as described in Section 3.4. The transit or ingress router uses the EAB address to program an IP route in the Routing Information Base (RIB) and uses the previously signaled EXPLICIT_ROUTE object to derive the next-hop information associated with the EAB route at that hop.
An advantage of using RSVP to establish IP-TE LSP tunnels is that it enables the allocation of resources along the path. For example, bandwidth can be allocated to each IP-TE LSP tunnel using standard RSVP reservations as described in [RFC3209].

### 3.6. Protection

Fast Reroute (FRR) procedures that are defined in [RFC4090] describe the mechanisms for router along the LSP path to act as a Point of Local Repair (PLR) and reroute traffic and signaling of a protected RSVP-TE LSP onto a pre-established bypass tunnel in the event of a protected TE link or node failure.

Similar mechanisms can be employed for protecting IP-TE LSP tunnel(s) in IP network(s). An ingress or transit router acting as potential PLR can pre-establish bypass tunnel(s) that protect the primary IP-TE LSP tunnel against the protected link or downstream node failure.

Upon failure of the protected link, the traffic arriving over the protected IP-TE LSP on the PLR is automatically tunneled over the pre-established bypass IP-TE LSP tunnel and packets are forwarded towards the Merge Point (MP) router. At the MP router, the incoming IP packets are decapsulated exposing the original IP header of the protected IP-TE LSP tunnel. The packets are forwarded downstream of the MP router along the

### 3.7. Shared Forwarding

One capability of the IP data plane is its ability to reuse the IP forwarding entry when setting up IP-TE LSP(s) from multiple sources and that share a common destination. This capability MAY be preserved provided certain requirements are met. We refer to this capability as "Shared Forwarding". Shared Forwarding is a local policy local to egress router responsible for binding an EAB address to the signaled IP-TE LSP tunnel.

The Shared Forwarding function allows the reduction of forwarding entries on any transit router RIB. The Shared forwarding paths are identical in function to independently routed Multi-point to Point (MP2P) paths that share part of their path(s) from the intersecting router and towards the egress router.

If the egress router policy allows for Shared Forwarding, and upon signaling a new IP-TE LSP tunnel, the egress inspects the recorded path (extracted from the RECORD_ROUTE object). If the egress router determines that the newly signaled IP-TE LSP path intersects and merges with other IP-TE LSP from the intersection point to the
egress, and if Shared Forwarding is enabled, it MUST assign the same EAB address bound to the existing IP-TE LSP tunnel.

Note, forwarding memory savings from Shared Forwarding can be quite dramatic in some topologies where a high degree of meshing is required.

3.8. Error Conditions

This section will be updated in future revisions of this document.

4. IANA Considerations

This section will be updated in future revisions of this document.

5. Security Considerations

This section will be updated in future revisions of this document.

6. Acknowledgement

The authors would like to thank Igor Bryskin for providing valuable feedback to this document.

7. Contributors

Raveendra Torvi
Juniper Networks

Email: rtorvi@juniper.net

8. References

8.1. Normative References


8.2. Informative References


Saad & Beeram            Expires January 9, 2020               [Page 10]
Authors’ Addresses

Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

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