

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: 11 August 2022

T. Saad
Juniper Networks
R. Gandhi
Cisco Systems Inc
X. Liu
Volta Networks
V.P. Beeram
Juniper Networks
I. Bryskin
Individual
O. Gonzalez de Dios
Telefonica
7 February 2022

A YANG Data Model for Traffic Engineering Tunnels, Label Switched Paths
and Interfaces
[draft-ietf-teas-yang-te-29](#)

Abstract

This document defines a YANG data model for the provisioning and management of Traffic Engineering (TE) tunnels, Label Switched Paths (LSPs), and interfaces. 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 11 August 2022.

Internet-Draft

TE YANG Data Model

February 2022

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the [Trust Legal Provisions](#) and are provided without warranty as described in the Revised BSD License.

Table of Contents

1.	Introduction	3
2.	Requirements Language	3
2.1.	Prefixes in Data Node Names	4
2.2.	Model Tree Diagrams	4
3.	Design Considerations	5
3.1.	State Data Organization	5
4.	Model Overview	6
4.1.	Module Relationship	6
5.	TE YANG Model	7
5.1.	Module Structure	7
5.1.1.	TE Globals	9
5.1.2.	TE Tunnels	12
5.1.3.	TE LSPs	19
5.2.	Tree Diagram	19
5.3.	YANG Module	60
6.	TE Device YANG Model	98
6.1.	Module Structure	99
6.1.1.	TE Interfaces	99
6.2.	Tree Diagram	100
6.3.	YANG Module	102
7.	Notifications	116
8.	TE Generic and Helper YANG Modules	117
9.	IANA Considerations	117
10.	Security Considerations	117
11.	Acknowledgement	119
12.	Contributors	119
13.	Appendix A: Data Tree Examples	119

13.1.	Basic Tunnel Setup	120
13.2.	Global Named Path Constraints	121
13.3.	Tunnel with Global Path Constraint	121
13.4.	Tunnel with Per-tunnel Path Constraint	122
13.5.	Tunnel State	123

14.	References	124
14.1.	Normative References	124
14.2.	Informative References	127
	Authors' Addresses	128

[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 Traffic Engineering (TE) tunnels, Label Switched Paths (LSPs), and interfaces. The model 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) [[I-D.ietf-spring-segment-routing-policy](#)] will augment the generic TE YANG module.

[2.](#) Requirements Language

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 [[RFC6241](#)] and are used in this specification:

- * client
- * configuration data

- * state data

This document also makes use of the following terminology introduced in the YANG Data Modeling Language [[RFC7950](#)]:

- * augment
- * data model
- * data node

[2.1](#). Prefixes in Data Node Names

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

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
rt-types	ietf-routing-types	[RFC8294]
te-types	ietf-te-types	[RFC8776]
te-packet-types	ietf-te-packet-types	[RFC8776]

+-----+-----+-----+			
te	ietf-te	this document	
+-----+-----+-----+			
te-dev	ietf-te-device	this document	
+-----+-----+-----+			

Table 1: Prefixes and corresponding YANG modules

2.2. Model Tree Diagrams

The tree diagrams extracted from the module(s) defined in this document are given in subsequent sections as per the syntax defined in [[RFC8340](#)].

3. Design Considerations

This document describes a generic TE 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 generic TE 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 generic TE YANG data model does not cover signaling protocol data. The signaling protocol used to instantiate TE LSPs are outside the scope of this document and expected to be covered by augmentations defined in other document(s).

The following other design considerations are taken into account with respect data organization:

- * The generic TE YANG data model 'ietf-te' contains device

independent data and can be used to model data off a device (e.g. on a TE 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.
- * Suitable defaults are specified for all configurable elements.
- * The model declares a number of TE functions as features that can be optionally supported.

[3.1.](#) State Data Organization

The Network Management Datastore Architecture (NMDA) [[RFC8342](#)] addresses modeling state data for ephemeral objects. This document adopts the NMDA model for configuration and state data representation as per IETF guidelines for new IETF YANG models.

[4.](#) Model Overview

The data models defined in this document cover the core TE features that are commonly supported by different vendor implementations. The support of extended or vendor specific TE feature(s) is expected to be in either augmentations, or deviations to the model defined in this document.

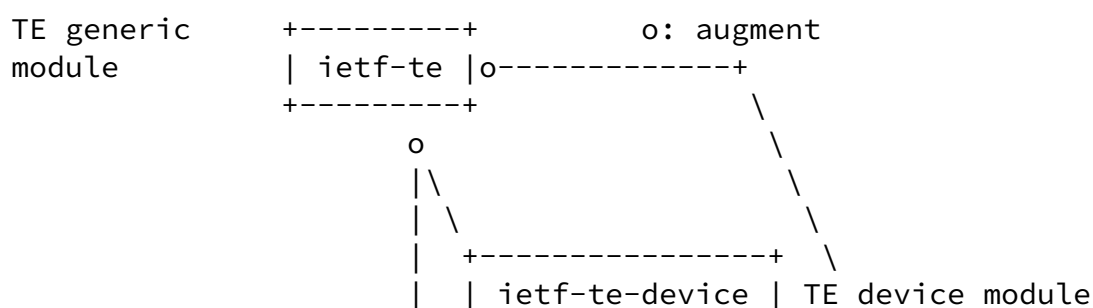
[4.1.](#) Module Relationship

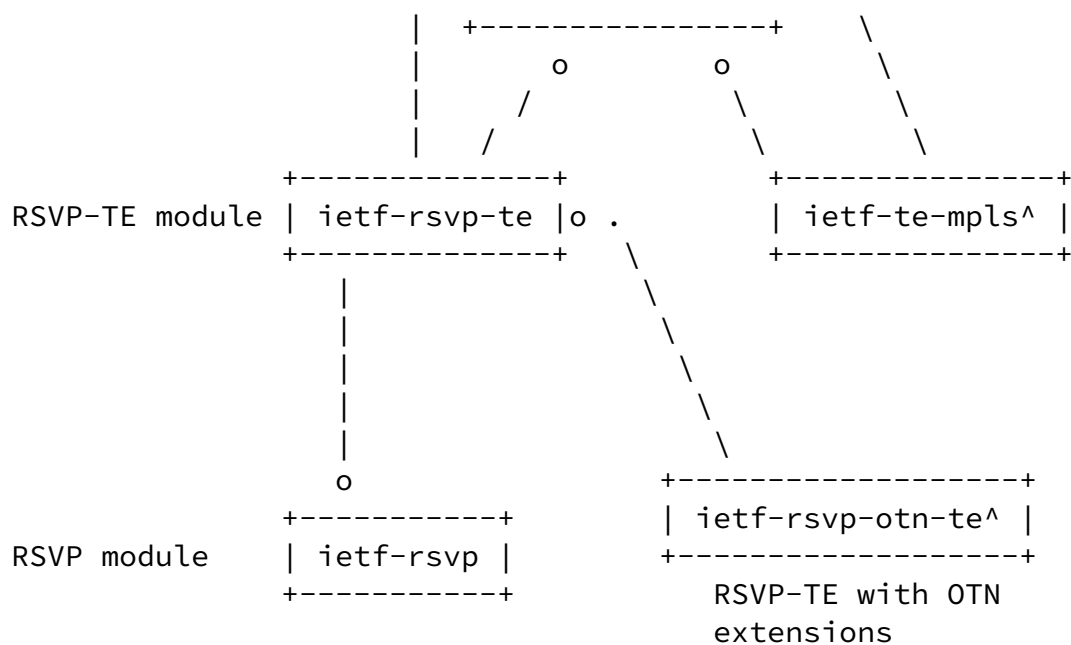
The generic TE YANG data model that is defined in "ietf-te.yang" covers the building blocks that are device independent and agnostic of any specific technology or control plane instances. The TE device model defined in "ietf-te-device.yang" augments the generic TE YANG data model and covers data that 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 generic TE 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](#)].





^ shown for illustration
(not in this document)

Figure 1: Relationship of TE module(s) with signaling protocol modules

5. TE YANG Model

The generic TE YANG module ('ietf-te') is meant to manage and operate a TE network. This includes creating, modifying and retrieving TE Tunnels, LSPs, and interfaces and their associated attributes (e.g. Administrative-Groups, SRLGs, etc.).

The detailed tree structure is provided in Figure 2.

5.1. Module Structure

The 'ietf-te' uses three main containers grouped under the main 'te' container (see Figure 2). The 'te' container is the top level container in the data model. The presence of the 'te' container enables TE function system wide. Below provides further descriptions of containers that exist under the 'te' top level container.

The 'globals' container maintains the set of global TE attributes that can be applicable to TE Tunnel(s) and interface(s).

tunnels:

The 'tunnels' container includes the list of TE Tunnels that are instantiated. Refer to [Section 5.1.2](#) for further details on the properties of a TE Tunnel.

lsps:

The 'lsps' container includes the list of TE LSP(s) that are instantiated for TE Tunnels. Refer to [Section 5.1.3](#) for further details on the properties of a TE LSP.

tunnels-path-compute:

A Remote Procedure Call (RPC) to request path computation for a specific TE Tunnel. The RPC allows requesting path computation using atomic and stateless operation. A tunnel may also be configured in 'compute-only' mode to provide stateful path updates - see [Section 5.1.2](#) for further details.

tunnels-action:

An RPC to request a specific action (e.g. reoptimize, or tear-and-setup) to be taken on a specific tunnel or all tunnels.

```
module: ietf-te
  +--rw te!
    +--rw globals
      .
      .
    +--rw tunnels
      .
      .
    +-- lsps
```

```
rpcs:
  +---x tunnels-path-compute
  +---x tunnels-action
```

Figure 2: TE Tunnel model high-level YANG tree view

[5.1.1.](#) TE Globals

The 'globals' container covers properties that control TE features behavior system-wide, and its respective state (see Figure 3). The TE globals configuration include:

```
+--rw globals
|   +--rw named-admin-groups
|   |   +--rw named-admin-group* [name]
|   ..
|   +--rw named-srlgs
|   |   +--rw named-srlg* [name] {te-types:named-srlg-groups}?
|   ..
|   +--rw named-path-constraints
|   |   +--rw named-path-constraint* [name]
|   ..
```

Figure 3: TE globals YANG subtree high-level structure

named-admin-groups:

A YANG container for the list of named (extended) administrative groups that may be applied to TE links.

named-srlgs:

A YANG container for the list named Shared Risk Link Groups (SRLGs) that may be applied to TE links.

named-path-constraints:

A YANG container for a list of named path constraints. Each named path constraint is composed of a set of constraints that can be applied during path computation. A named path constraint can be applied to multiple TE Tunnels. Path constraints may also be specified directly under the TE Tunnel. The path constraint specified under the TE Tunnel take precedence over the path constraints derived from the referenced named path constraint. A named path constraint entry can be formed up of the following path constraints:

```

|  +--rw named-path-constraints
|    +--rw named-path-constraint* [name]
|      +--rw name                                string
|      +--rw te-bandwidth
// ...
|      +--rw link-protection?                    identityref
|      +--rw setup-priority?                      uint8
|      +--rw hold-priority?                      uint8
|      +--rw signaling-type?                    identityref
|      +--rw path-metric-bounds
// ...
|      +--rw path-affinities-values
// ...
|      +--rw path-affinity-names
// ...
|      +--rw path-srlgs-lists
// ...
|      +--rw path-srlgs-names
// ...
|      +--rw disjointness?
|        |      te-path-disjointness
// ...
|      +--rw explicit-route-objects-always
// ...
|        |  +--rw route-object-exclude-always* [index]
|
|        |  +--rw route-object-include-exclude* [index]

```

Figure 4: Named path constraints YANG subtree

- o te-bandwidth: A YANG container that holds the technology agnostic TE bandwidth constraint.
- o link-protection: A YANG leaf that holds the link protection type constraint required for the links to be included in the computed path.
- o setup/hold priority: A YANG leaf that holds the LSP setup and hold admission priority as defined in [\[RFC3209\]](#).

- o signaling-type: A YANG leaf that holds the LSP setup type, such as RSVP-TE or SR.
- o path-metric-bounds: A YANG container that holds the set of metric bounds applicable on the computed TE tunnel path.

- o path-affinities-values: A YANG container that holds the set of affinity values and mask to be used during path computation.
- o path-affinity-names: A YANG container that holds the set of named affinity constraints and corresponding inclusion or exclusions instruction for each to be used during path computation.
- o path-srlgs-lists: A YANG container that holds the set of SRLG values and corresponding inclusion or exclusions instruction to be used during path computation.
- o path-srlgs-names: A YANG container that holds the set of named SRLG constraints and corresponding inclusion or exclusions instruction for each to be used during path computation.
- o disjointness: The level of resource disjointness constraint that the secondary path of a TE tunnel has to adhere to.
- o explicit-route-objects-always: A YANG container that contains two route objects lists:
 - + 'route-object-exclude-always': a list of route entries to always exclude from the path computation.
 - + 'route-object-include-exclude': a list of route entries to include or exclude in the path computation.

The 'route-object-include-exclude' is used to configure constraints on which route objects (e.g., nodes, links) are

included or excluded in the path computation.

The interpretation of an empty 'route-object-include-exclude' list depends on the TE Tunnel (end-to-end or Tunnel Segment) and on the specific path, according to the following rules:

1. An empty 'route-object-include-exclude' list for the primary path of an end-to-end TE Tunnel indicates that there are no route objects to be included or excluded in the path computation.
2. An empty 'route-object-include-exclude' list for the primary path of a TE Tunnel Segment indicates that no primary LSP is required for that TE Tunnel.

3. An empty 'route-object-include-exclude' list for a reverse path means it always follows the forward path (i.e., the TE Tunnel is co-routed). When the 'route-object-include-exclude' list is not empty, the reverse path is routed independently of the forward path.
4. An empty 'route-object-include-exclude' list for the secondary (forward) path indicates that the secondary path has the same endpoints as the primary path.

[5.1.2.](#) TE Tunnels

The 'tunnels' container holds the list of TE Tunnels that are provisioned on devices in the network (see Figure 5).

A TE Tunnel in the list is uniquely identified by a name. When the model is used to manage a specific device, the 'tunnels' list contains the TE Tunnels originating from the specific device. When the model is used to manage a TE controller, the 'tunnels' list contains all TE Tunnels and TE tunnel segments originating from device(s) that the TE controller manages.

The TE Tunnel model allows the configuration and management of the following TE tunnel related objects:

TE Tunnel:

A YANG container of one or more LSPs established between the source and destination TE Tunnel termination points. A TE Tunnel LSP is a connection-oriented service provided by the network layer for the delivery of client data between a source and the destination of the TE Tunnel termination points.

TE Tunnel Segment:

A part of a multi-domain TE Tunnel that is within a specific network domain.

```
+--rw tunnels
|  +--rw tunnel* [name]
|    +--rw name                string
|    +--rw alias?              string
|    +--rw identifier?         uint32
|    +--rw color?              uint32
|    +--rw description?        string
|    +--ro operational-state?   identityref
|    +--rw encoding?           identityref
|    +--rw switching-type?     identityref
|    +--rw admin-state?        identityref
|    +--rw reoptimize-timer?   uint16
|    +--rw source?             te-types:te-node-id
|    +--rw destination?        te-types:te-node-id
|    +--rw src-tunnel-tp-id?   binary
|    +--rw dst-tunnel-tp-id?   binary
|    +--rw controller
|      |  +--rw protocol-origin? identityref
```

```

|      |  +--rw controller-entity-id?      string
|      +--rw bidirectional?                boolean
|      +--rw association-objects
|      |  +--rw association-object* [association-key]
// ..
|      |
|      +--rw protection
// ..
|      +--rw restoration
// ..
|      +--rw te-topology-identifier
// ..
|      +--rw hierarchy
// ..

```

Figure 5: TE Tunnel list YANG subtree structure

The TE Tunnel has a number of attributes that are set directly under the tunnel (see Figure 5). The main attributes of a TE Tunnel are described below:

operational-state:

A YANG leaf that holds the operational state of the tunnel.

name:

A YANG leaf that holds the name of a TE Tunnel. The name of the TE Tunnel uniquely identifies the tunnel within the TE tunnel list. The name of the TE Tunnel can be formatted as a Uniform

Resource Indicator (URI) by including the namespace to ensure uniqueness of the name amongst all the TE Tunnels present on devices and controllers.

alias:

A YANG leaf that holds an alternate name to the TE tunnel. Unlike the TE tunnel name, the alias can be modified at any time during the lifetime of the TE tunnel.

identifier:

A YANG leaf that holds an identifier of the tunnel. This identifier is unique amongst tunnels originated from the same ingress device.

color:

A YANG leaf that holds the color associated with the TE tunnel. The color is used to map or steer services that carry matching color on to the TE tunnel as described in [[RFC9012](#)].

encoding/switching:

The 'encoding' and 'switching-type' are YANG leaves that define the specific technology in which the tunnel operates in as described in [[RFC3945](#)].

reoptimize-timer:

A YANG leaf to set the interval period for tunnel reoptimization.

source/destination:

YANG leaves that define the tunnel source and destination node endpoints.

src-tunnel-tp-id/dst-tunnel-tp-id:

YANG leaves that hold the identifiers of source and destination TE Tunnel Termination Points (TTPs) [[RFC8795](#)] residing on the source and destination nodes. The TTP identifiers are optional on nodes that have a single TTP per node. For example, TTP identifiers are optional for packet (IP/MPLS) routers.

controller:

A YANG container that holds tunnel data relevant to an optional external TE controller that may initiate or control a tunnel. This target node may be augmented by external module(s), for example, to add data for PCEP initiated and/or delegated tunnels.

bidirectional:

A YANG leaf that when present indicates the LSPs of a TE Tunnel are bidirectional and co-routed.

association-objects:

A YANG container that holds the set of associations of the TE Tunnel to other TE Tunnels. Associations at the TE Tunnel level apply to all paths of the TE Tunnel. The TE tunnel associations can be overridden by associations configured directly under the TE Tunnel path.

protection:

A YANG container that holds the TE Tunnel protection properties.

restoration:

A YANG container that holds the TE Tunnel restoration properties.

te-topology-identifier:

A YANG container that holds the topology identifier associated with the topology where paths for the TE tunnel are computed.

```
+--rw hierarchy
|  +--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
```

Figure 6: TE Tunnel hierarchy YANG subtree

hierarchy:

A YANG container that holds hierarchy related properties of the TE Tunnel (see Figure 6. A TE LSP can be set up in MPLS or Generalized MPLS (GMPLS) networks to be used as a TE links to carry traffic in other (client) networks [[RFC6107](#)]. In this case, the model introduces the TE Tunnel hierarchical link endpoint parameters to identify the specific link in the client layer that the underlying TE Tunnel is associated with. The hierarchy container includes the following:

- o dependency-tunnels: A set of hierarchical TE Tunnels provisioned or to be provisioned in the immediate lower layer that this TE tunnel depends on for multi-layer path computation. A dependency TE Tunnel is provisioned if and only if it is used (selected by path computation) at least by one client layer TE Tunnel. The TE link in the client layer network topology supported by a dependent TE Tunnel is dynamically created only when the dependency TE Tunnel is actually provisioned.
- o hierarchical-link: A YANG container that holds the identity of the hierarchical link (in the client layer) that is supported by this TE Tunnel. The endpoints of the hierarchical link are defined by TE tunnel source and destination node endpoints. The hierarchical link can be identified by its source and destination link termination point identifiers.

[5.1.2.1](#). TE Tunnel Paths

The TE Tunnel can be configured with a set of paths that define the tunnel forward and reverse paths as described in Figure 7. Moreover, a primary path can be specified a set of candidate secondary paths that can be visited to support path protection. The following describe further the list of paths associated with a TE Tunnel.

Internet-Draft

TE YANG Data Model

February 2022

```

|      +--rw primary-paths
|      |      +--rw primary-path* [name]
|      |      +--rw name                                     string
// ..
|      |      +
|      |      +--rw primary-reverse-path
|      |      |      +--rw name?                             string
// ..
|      |      |      |
|      |      |      |      +--rw candidate-secondary-reverse-paths
|      |      |      |      |      +--rw candidate-secondary-reverse-path*
|      |      |      |      |      [secondary-path]
|      |      |      |      |      +--rw secondary-path      leafref
|      |      |      |      +--rw candidate-secondary-paths
|      |      |      |      |      +--rw candidate-secondary-path* [secondary-path]
|      |      |      |      |      +--rw secondary-path      leafref
|      |      |      |      +--ro active?                    boolean
|      |      |
|      |      +--rw secondary-paths
|      |      |      +--rw secondary-path* [name]
|      |      |      +--rw name                                     string
// ..
|      |      +--rw secondary-reverse-paths
|      |      |      +--rw secondary-reverse-path* [name]
|      |      |      +--rw name                                     string

```

Figure 7: TE Tunnel paths YANG tree structure

primary-paths:

A YANG container that holds the list of primary paths. A primary path is identified by 'name'. A primary path is selected from the list to instantiate a primary forwarding LSP for the tunnel. The list of primary paths is visited by order of preference. A primary path has the following attributes:

- **primary-reverse-path:** A YANG container that holds properties of the primary reverse path. The reverse path is applicable to bidirectional TE Tunnels.

- candidate-secondary-paths: A YANG container that holds a list of candidate secondary paths which may be used for the primary path to support path protection. The candidate secondary path(s) reference path(s) from the tunnel secondary paths list. The preference of the secondary paths is specified within the list and dictates the order of visiting the secondary path from the list. The attributes of a secondary path can be defined

separately from the primary path. The attributes of a secondary path will be inherited from the associated 'active' primary when not explicitly defined for the secondary path.

secondary-paths:

A YANG container that holds the set of secondary paths. A secondary path is identified by 'name'. A secondary path can be referenced from the TE Tunnel's 'candidate-secondary-path' list. A secondary path contains attributes similar to a primary path.

secondary-reverse-paths:

A YANG container that holds the set of secondary reverse paths. A secondary reverse path is identified by 'name'. A secondary reverse path can be referenced from the TE Tunnel's 'candidate-secondary-reverse-paths' list. A secondary reverse path contains attributes similar to a primary path.

The following set common path attributes are shared for primary forward and reverse primary and secondary paths:

compute-only:

A path of TE Tunnel is, by default, provisioned so that it can be instantiated in forwarding to carry traffic as soon as a valid path is computed. In some cases, a TE path 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 path is configured in 'compute-only' mode to distinguish it from the default behavior. A 'compute-only' path is configured as a usual with the associated per path constraint(s) and

properties on a device or TE controller. The device or TE controller computes the feasible path(s) subject to configured constraints. A client may query the 'compute-only' computed path properties 'on-demand', or alternatively, can subscribe to be notified of computed path(s) and whenever the path properties change.

use-path-computation:

A YANG leaf that indicates whether or not path computation is to be used for a specified path.

lockdown:

A YANG leaf that when set indicates the existing path should not be reoptimized after a failure on any of its traversed links.

te-topology-identifier:

A YANG container that holds the topology identifier associated with the tunnel.

optimizations:

a YANG container that holds the optimization objectives that path computation will use to select a path.

computed-paths-properties: > A YANG container that holds properties for the list of computed paths.

computed-path-error-infos:

A YANG container that holds a list of errors related to the path.

lsps:

a YANG container that holds a list of LSPs that are instantiated for this specific path.

[5.1.3.](#) TE LSPs

The 'lsps' container includes the set of TE LSP(s) that are instantiated. A TE LSP is identified by a 3-tuple ('tunnel-name', 'node', 'lsp-id').

When the model is used to manage a specific device, the 'lsps' list contains all TE LSP(s) that traverse the device (including ingressing, transiting and egressing the device).

When the model is used to manage a TE controller, the 'lsps' list contains all TE LSP(s) that traverse all network devices (including ingressing, transiting and egressing the device) that the TE controller manages.

[5.2.](#) Tree Diagram

Figure 8 shows the tree diagram of the generic TE YANG model defined in modules 'ietf-te.yang'.

```
module: ietf-te
  +--rw te!
    +--rw globals
      | +--rw named-admin-groups
      | | +--rw named-admin-group* [name]
      | | {te-types:extended-admin-groups,te-types:named-extend
ed-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 value? te-types:srlg
      | | +--rw cost? uint32
      | +--rw named-path-constraints
      | | +--rw named-path-constraint* [name]
      | | {te-types:named-path-constraints}?
      | | +--rw name string
      | | +--rw te-bandwidth
```



```

|         +---rw hop-type?      te-hop-type
|         +---rw direction?    te-link-direction
+---:(unnumbered-link-hop)
|   +---rw unnumbered-link-hop
|       +---rw link-tp-id      te-tp-id
|       +---rw node-id        te-node-id
|       +---rw hop-type?      te-hop-type
|       +---rw direction?    te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|       +---rw as-number      inet:as-number
|       +---rw hop-type?      te-hop-type
+---:(label)
|   +---rw label-hop
|       +---rw te-label
|           +---rw (technology)?
|               |   +---:(generic)
|               |       +---rw generic?
|               |           rt-types:generalized-label
|           +---rw direction?
|               te-label-direction
+---rw route-object-include-exclude* [index]
+---rw explicit-route-usage?      identityref
+---rw index                      uint32
+---rw (type)?
+---:(numbered-node-hop)
|   +---rw numbered-node-hop
|       +---rw node-id      te-node-id
|       +---rw hop-type?    te-hop-type
+---:(numbered-link-hop)
|   +---rw numbered-link-hop
|       +---rw link-tp-id    te-tp-id
|       +---rw hop-type?    te-hop-type
|       +---rw direction?    te-link-direction
+---:(unnumbered-link-hop)
|   +---rw unnumbered-link-hop

```

```

|         +---rw link-tp-id      te-tp-id
|         +---rw node-id        te-node-id
|         +---rw hop-type?      te-hop-type
|         +---rw direction?    te-link-direction
+---:(as-number)

```



```

|   +---rw as-number-hop
|   +---rw as-number      inet:as-number
|   +---rw hop-type?      te-hop-type
+---:(label)
|   +---rw label-hop
|   +---rw te-label
|       +---rw (technology)?
|           |   +---:(generic)
|           |       +---rw generic?
|           |           rt-types:generalized-label
|       +---rw direction?
|                   te-label-direction
+---:(srlg)
|   +---rw srlg
|       +---rw srlg?      uint32
+---rw path-in-segment!
|   +---rw label-restrictions
|       +---rw label-restriction* [index]
|       +---rw restriction?      enumeration
|       +---rw index              uint32
+---rw label-start
|   +---rw te-label
|       +---rw (technology)?
|           |   +---:(generic)
|           |       +---rw generic?
|           |           rt-types:generalized-label
|       +---rw direction?
|                   te-label-direction
+---rw label-end
|   +---rw te-label
|       +---rw (technology)?
|           |   +---:(generic)
|           |       +---rw generic?
|           |           rt-types:generalized-label
|       +---rw direction?
|                   te-label-direction
+---rw label-step
|   +---rw (technology)?
|       +---:(generic)
|           +---rw generic?      int32
+---rw range-bitmap?      yang:hex-string
+---rw path-out-segment!
|   +---rw label-restrictions

```

```

        +---rw label-restriction* [index]
            +---rw restriction?      enumeration
            +---rw index              uint32
            +---rw label-start
            |   +---rw te-label
            |   |   +---rw (technology)?
            |   |   |   +---:(generic)
            |   |   |   +---rw generic?
            |   |   |       rt-types:generalized-label
            |   |   +---rw direction?
            |   |       te-label-direction
            +---rw label-end
            |   +---rw te-label
            |   |   +---rw (technology)?
            |   |   |   +---:(generic)
            |   |   |   +---rw generic?
            |   |   |       rt-types:generalized-label
            |   |   +---rw direction?
            |   |       te-label-direction
            +---rw label-step
            |   +---rw (technology)?
            |   |   +---:(generic)
            |   |   +---rw generic?    int32
            +---rw range-bitmap?      yang:hex-string
+---rw tunnels
    +---rw tunnel* [name]
        +---rw name                    string
        +---rw alias?                  string
        +---rw identifier?              uint32
        +---rw color?                   uint32
        +---rw description?              string
        +---rw admin-state?              identityref
        +---ro operational-state?         identityref
        +---rw encoding?                  identityref
        +---rw switching-type?             identityref
        +---rw source?                    te-types:te-node-id
        +---rw destination?               te-types:te-node-id
        +---rw src-tunnel-tp-id?           binary
        +---rw dst-tunnel-tp-id?           binary
        +---rw bidirectional?              boolean
        +---rw controller
            |   +---rw protocol-origin?      identityref
            |   +---rw controller-entity-id? string
        +---rw reoptimize-timer?           uint16
        +---rw association-objects
            |   +---rw association-object* [association-key]
            |   |   +---rw association-key    string
            |   |   +---rw type?              identityref

```

Internet-Draft

TE YANG Data Model

February 2022

```

| | | +--rw id?                               uint16
| | | +--rw source
| | | |   +--rw id?       te-gen-node-id
| | | |   +--rw type?    enumeration
| | | +--rw association-object-extended* [association-key]
| | | |   +--rw association-key    string
| | | |   +--rw type?             identityref
| | | |   +--rw id?               uint16
| | | |   +--rw source
| | | | |   +--rw id?       te-gen-node-id
| | | | |   +--rw type?    enumeration
| | | | +--rw global-source?    uint32
| | | | +--rw extended-id?      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
| +--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 hierarchy
| |   +--rw dependency-tunnels

```

```

| | | +--rw dependency-tunnel* [name]
| | |   +--rw name
| | |   |       -> /te/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 primary-paths
| | +--rw primary-path* [name]
| | | +--rw name                                string
| | | +--rw path-computation-method?           identityref
| | | +--rw path-computation-server
| | | | +--rw id?      te-gen-node-id
| | | | +--rw type?   enumeration
| | | +--rw compute-only?                       empty
| | | +--rw use-path-computation?               boolean
| | | +--rw lockdown?                          empty
| | | +--ro path-scope?                        identityref
| | | +--rw preference?                        uint8
| | | +--rw k-requested-paths?                 uint8
| | | +--rw association-objects
| | | | +--rw association-object* [association-key]
| | | | | +--rw association-key    string
| | | | | +--rw type?              identityref
| | | | | +--rw id?                uint16
| | | | | +--rw source
| | | | | | +--rw id?      te-gen-node-id
| | | | | | +--rw type?   enumeration
| | | | +--rw association-object-extended*
| | | | | [association-key]
| | | | | +--rw association-key    string
| | | | | +--rw type?              identityref
| | | | | +--rw id?                uint16
| | | | | +--rw source
| | | | | | +--rw id?      te-gen-node-id

```


					+--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:ge
neralized-label					
					+--rw direction?
					te-label-directio
n					
					+--rw tiebreakers

```
| | | | +--rw tiebreaker* [tiebreaker-type]
| | | | +--rw tiebreaker-type identityref
| | | | +--:(objective-function)
| | | | {path-optimization-objective-function}?
| | | | +--rw objective-function
| | | | +--rw objective-function-type?
| | | | identityref
| | | | +--rw named-path-constraint? leafref
| | | | {te-types:named-path-constraints}?
| | | | +--rw te-bandwidth
| | | | | +--rw (technology)?
| | | | | +--:(generic)
| | | | | +--rw generic? te-bandwidth
| | | | +--rw link-protection? identityref
```


bel

```

    +--:(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 path-in-segment!
|   +--rw label-restrictions
|       +--rw label-restriction* [index]
|           +--rw restriction?    enumeration
|           +--rw index            uint32
|           +--rw label-start
|               +--rw te-label
|                   +--rw (technology)?
|                       +--:(generic)
|                           +--rw generic?
|                               rt-types:generalized-label
|                   +--rw direction?
|                       te-label-direction
|           +--rw label-end
|               +--rw te-label
|                   +--rw (technology)?
|                       +--:(generic)
|                           +--rw generic?
|                               rt-types:generalized-label
|                   +--rw direction?
|                       te-label-direction
|           +--rw label-step
|               +--rw (technology)?
|                   +--:(generic)
|                       +--rw generic?    int32
|           +--rw range-bitmap?    yang:hex-string
+--rw path-out-segment!
|   +--rw label-restrictions
|       +--rw label-restriction* [index]
|           +--rw restriction?    enumeration
|           +--rw index            uint32
|           +--rw label-start
|               +--rw te-label
|                   +--rw (technology)?
|                       +--:(generic)

```

```
| | | +---rw generic?
| | | | rt-types:generalized-label
| | | +---rw direction?
| | | | te-label-direction
+---rw label-end
| | +---rw te-label
| | | +---rw (technology)?
| | | | +---:(generic)
| | | | | +---rw generic?
| | | | | | rt-types:generalized-label
| | | | +---rw direction?
| | | | | te-label-direction
+---rw label-step
| | +---rw (technology)?
| | | +---:(generic)
| | | | +---rw generic? int32
+---rw range-bitmap? yang:hex-string
+--ro computed-paths-properties
| +--ro computed-path-properties* [k-index]
| +--ro k-index uint8
| +--ro path-properties
| | +--ro path-metric* [metric-type]
| | | +--ro metric-type identityref
| | | +--ro accumulative-value? uint64
+--ro path-affinities-values
| | +--ro path-affinities-value* [usage]
| | | +--ro usage identityref
| | | +--ro value? admin-groups
+--ro path-affinity-names
| | +--ro path-affinity-name* [usage]
| | | +--ro usage identityref
| | | +--ro affinity-name* [name]
| | | | +--ro name string
+--ro path-srlgs-lists
| | +--ro path-srlgs-list* [usage]
| | | +--ro usage identityref
| | | +--ro values* srlg
+--ro path-srlgs-names
| | +--ro path-srlgs-name* [usage]
| | | +--ro usage identityref
| | | +--ro names* string
+--ro path-route-objects
| | +--ro path-route-object* [index]
```

				++-ro index
				uint32
				++-ro (type)?
				+++:(numbered-node-hop)
				++-ro numbered-node-hop

				+++ro node-id	te-node-id
				+++ro hop-type?	
					te-hop-type
				+++:(numbered-link-hop)	
				+++ro numbered-link-hop	
				+++ro link-tp-id	te-tp-id
				+++ro hop-type?	
					te-hop-type
				+++ro direction?	
					te-link-direction
				+++:(unnumbered-link-hop)	
				+++ro unnumbered-link-hop	
				+++ro link-tp-id	te-tp-id
				+++ro node-id	
					te-node-id
				+++ro hop-type?	
					te-hop-type
				+++ro direction?	
					te-link-direction
				+++:(as-number)	
				+++ro as-number-hop	
				+++ro as-number	
					inet:as-number
				+++ro hop-type?	
					te-hop-type
				+++:(label)	
				+++ro label-hop	
				+++ro te-label	
				+++ro (technology)?	
					+++:(generic)
					+++ro generic?
					rt-types:gener
				+++ro direction?	
					te-label-direction
				+++ro te-bandwidth	

alized-label

}?					{path-optimization-objective-function
					<pre> +---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 +---rw path-metric-bounds +---rw path-metric-bound* [metric-type] +---rw metric-type identityref +---rw upper-bound? uint64 +---rw path-affinities-values +---rw path-affinities-value* [usage] +---rw usage identityref +---rw value? admin-groups +---rw path-affinity-names +---rw path-affinity-name* [usage] +---rw usage identityref +---rw affinity-name* [name] +---rw name string +---rw path-srlgs-lists +---rw path-srlgs-list* [usage] +---rw usage identityref +---rw values* srlg +---rw path-srlgs-names +---rw path-srlgs-name* [usage] +---rw usage identityref +---rw names* string +---rw disjointness? te-path-disjointness </pre>

			+---rw explicit-route-objects-always
--	--	--	--------------------------------------

				<pre> +---rw route-object-exclude-always* [index] +---rw index uint32 +---rw (type)? +---:(numbered-node-hop) +---rw numbered-node-hop +---rw node-id te-node-id +---rw hop-type? te-hop-type +---:(numbered-link-hop) +---rw numbered-link-hop +---rw link-tp-id te-tp-id +---rw hop-type? te-hop-type +---rw direction? te-link-direction +---:(unnumbered-link-hop) +---rw unnumbered-link-hop +---rw link-tp-id te-tp-id +---rw node-id te-node-id +---rw hop-type? te-hop-type +---rw direction? te-link-direction +---:(as-number) +---rw as-number-hop +---rw as-number inet:as-number +---rw hop-type? te-hop-type +---:(label) +---rw label-hop +---rw te-label +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized +---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 </pre>
-label				

					<pre> +---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 </pre>
-label					<pre> +---rw direction? te-label-direction +---:(srlg) +---rw srlg +---rw srlg? uint32 +---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-la </pre>
bel					<pre> +---rw direction? te-label-direction +---rw label-end +---rw te-label +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized-la </pre>

bel

					+++rw direction?
--	--	--	--	--	------------------

Saad, et al.

Expires 11 August 2022

[Page 38]

Internet-Draft

TE YANG Data Model

February 2022

					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-la
bel					
					+++rw direction?
					te-label-direction
				+++rw label-end	
				+++rw te-label	
				+++rw (technology)?	
				+++:(generic)	
				+++rw generic?	
					rt-types:generalized-la
bel					
					+++rw direction?
					te-label-direction
				+++rw label-step	
				+++rw (technology)?	
				+++:(generic)	
				+++rw generic? int32	
				+++rw range-bitmap? yang:hex-string	
				+++ro computed-paths-properties	
				+++ro computed-path-properties* [k-index]	
				+++ro k-index uint8	
				+++ro path-properties	
				+++ro path-metric* [metric-type]	

					<pre> +---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] </pre>
--	--	--	--	--	---

					<pre> +---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-route-object* [index] +---ro index uint32 +---ro (type)? +---:(numbered-node-hop) +---ro numbered-node-hop +---ro node-id te-node-id +---ro hop-type? te-hop-type +---:(numbered-link-hop) +---ro numbered-link-hop +---ro link-tp-id te-tp-id +---ro hop-type? te-hop-type +---ro direction? te-link-direction +---:(unnumbered-link-hop) +---ro unnumbered-link-hop </pre>
--	--	--	--	--	---


```

        [association-key]
+--rw association-key      string
+--rw type?                identityref
+--rw id?                  uint16
+--rw source
|   +--rw id?              te-gen-node-id
|   +--rw type?            enumeration
+--rw global-source?       uint32
+--rw extended-id?         yang:hex-string
+--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

```



```

|         +---:(generic)
|         |         +---rw generic?    te-bandwidth
+---rw link-protection?                identityref
+---rw setup-priority?                 uint8
+---rw hold-priority?                  uint8
+---rw signaling-type?                 identityref
+---rw path-metric-bounds
|   +---rw path-metric-bound* [metric-type]
|   +---rw metric-type        identityref
|   +---rw upper-bound?      uint64
+---rw path-affinities-values
|   +---rw path-affinities-value* [usage]
|   +---rw usage                identityref
|   +---rw value?               admin-groups
+---rw path-affinity-names
|   +---rw path-affinity-name* [usage]
|   +---rw usage                identityref
|   +---rw affinity-name* [name]
|   +---rw name                 string
+---rw path-srlgs-lists
|   +---rw path-srlgs-list* [usage]
|   +---rw usage                identityref
|   +---rw values*             srlg
+---rw path-srlgs-names
|   +---rw path-srlgs-name* [usage]
|   +---rw usage                identityref
|   +---rw names*              string
+---rw disjointness?
|   te-path-disjointness
+---rw explicit-route-objects-always
|   +---rw route-object-exclude-always* [index]
|   |   +---rw index                uint32
|   |   +---rw (type)?
|   |   |   +---:(numbered-node-hop)
|   |   |   |   +---rw numbered-node-hop
|   |   |   |   |   +---rw 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
--	--	--	--	--	------------------	-------------------

bel

				+++:(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-la
				+++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

					<pre> +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized-la </pre>
bel					<pre> +---rw direction? te-label-direction +---:(srlg) </pre>

					<pre> +---rw srlg +---rw srlg? uint32 +---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 </pre>
--	--	--	--	--	---

```
| | | +--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 computed-path-properties* [k-index]  
| | | +--ro k-index uint8
```

```

+--ro path-properties
  +--ro path-metric* [metric-type]
    | +--ro metric-type          identityref
    | +--ro accumulative-value?  uint64
  +--ro path-affinities-values
    | +--ro path-affinities-value* [usage]
    |   +--ro usage          identityref
    |   +--ro value?        admin-groups
  +--ro path-affinity-names
    | +--ro path-affinity-name* [usage]
    |   +--ro usage          identityref
    |   +--ro affinity-name* [name]
    |     +--ro name        string
  +--ro path-srlgs-lists
    | +--ro path-srlgs-list* [usage]
    |   +--ro usage          identityref
    |   +--ro values*       srlg
  +--ro path-srlgs-names
    | +--ro path-srlgs-name* [usage]
    |   +--ro usage          identityref
    |   +--ro names*        string
+--ro path-route-objects

```

```

+--ro path-route-object* [index]
  +--ro index
    | uint32
  +--ro (type)?
    +--:(numbered-node-hop)
      | +--ro numbered-node-hop
      |   +--ro node-id      te-node-id
      |   +--ro hop-type?
      |     te-hop-type
    +--:(numbered-link-hop)
      | +--ro numbered-link-hop
      |   +--ro link-tp-id   te-tp-id
      |   +--ro hop-type?
      |     | te-hop-type
      |   +--ro direction?
      |     te-link-direction
    +--:(unnumbered-link-hop)
      | +--ro unnumbered-link-hop
      |   +--ro link-tp-id   te-tp-id

```



```

    +---ro lsp* [node lsp-id]
    +---ro tunnel-name?
    |         -> /te/lsp/lsp/tunnel-name
    +---ro node         -> /te/lsp/lsp/node
    +---ro lsp-id        -> /te/lsp/lsp/lsp-id
+---rw secondary-reverse-paths
|   +---rw secondary-reverse-path* [name]
|   |   +---rw name                                string
|   |   +---rw path-computation-method?            identityref
|   |   +---rw path-computation-server
|   |   |   +---rw id?      te-gen-node-id
|   |   |   +---rw type?    enumeration
|   |   +---rw compute-only?                        empty
|   |   +---rw use-path-computation?                boolean
|   |   +---rw lockdown?                            empty
|   |   +---ro path-scope?                          identityref
|   |   +---rw preference?                          uint8
|   +---rw association-objects
|   |   +---rw association-object* [association-key]
|   |   |   +---rw association-key    string
|   |   |   +---rw type?              identityref
|   |   |   +---rw id?                uint16
|   |   |   +---rw source
|   |   |   |   +---rw id?      te-gen-node-id
|   |   |   |   +---rw type?    enumeration
|   |   +---rw association-object-extended*
|   |   |   [association-key]
|   |   |   +---rw association-key    string
|   |   |   +---rw type?              identityref
|   |   |   +---rw id?                uint16
|   |   |   +---rw source
|   |   |   |   +---rw id?      te-gen-node-id
|   |   |   |   +---rw type?    enumeration
|   |   +---rw global-source?        uint32
|   |   +---rw extended-id?          yang:hex-string
+---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)

```

[illegible]

February 2022

```

+---rw (technology)?
|   +---:(generic)
|       +---rw generic?
|           rt-types:ge
neralized-label
|   |   |
|   |   |   +---rw direction?
|   |   |       te-label-directio
n
|   |   |   +---rw tiebreakers
|   |   |       +---rw tiebreaker* [tiebreaker-type]
|   |   |         +---rw tiebreaker-type    identityref
|   |   +---:(objective-function)
|   |       {path-optimization-objective-function}?
|   |       +---rw objective-function
|   |           +---rw objective-function-type?
|   |               identityref
+---rw named-path-constraint?          leafref
|   {te-types:named-path-constraints}?
+---rw te-bandwidth
|   +---rw (technology)?
|       +---:(generic)
|           +---rw generic?      te-bandwidth
+---rw link-protection?                  identityref
+---rw setup-priority?                    uint8
+---rw hold-priority?                     uint8
+---rw signaling-type?                   identityref
+---rw path-metric-bounds
|   +---rw path-metric-bound* [metric-type]
|       +---rw metric-type        identityref
|       +---rw upper-bound?     uint64
+---rw path-affinities-values
|   +---rw path-affinities-value* [usage]
|       +---rw usage              identityref
|       +---rw value?            admin-groups
+---rw path-affinity-names
|   +---rw path-affinity-name* [usage]
|       +---rw usage                identityref
|       +---rw affinity-name* [name]
|           +---rw name             string
+---rw path-srlgs-lists
|   +---rw path-srlgs-list* [usage]
|       +---rw usage                 identityref
```


				+++rw 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-la
					+++rw direction?
					te-label-direction
					+++:(srlg)
					+++rw srlg
					+++rw srlg? uint32
					+++rw path-in-segment!
					+++rw label-restrictions
					+++rw label-restriction* [index]
					+++rw restriction? enumeration
					+++rw index uint32
					+++rw label-start

					+--ro values* srlg
					+--ro path-srlgs-names
					+--ro path-srlgs-name* [usage]
					+--ro usage identityref
					+--ro names* string
					+--ro path-route-objects
					+--ro path-route-object* [index]
					+--ro index
					uint32
					+--ro (type)?
					+--:(numbered-node-hop)
					+--ro numbered-node-hop
					+--ro node-id te-node-id
					+--ro hop-type?
					te-hop-type
					+--:(numbered-link-hop)
					+--ro numbered-link-hop
					+--ro link-tp-id te-tp-id
					+--ro hop-type?
					te-hop-type
					+--ro direction?
					te-link-direction
					+--:(unnumbered-link-hop)
					+--ro unnumbered-link-hop
					+--ro link-tp-id te-tp-id
					+--ro node-id
					te-node-id
					+--ro hop-type?
					te-hop-type
					+--ro direction?
					te-link-direction
					+--:(as-number)
					+--ro as-number-hop
					+--ro as-number

						inet:as-number
						+--ro hop-type?
						te-hop-type
						+--:(label)
						+--ro label-hop
						+--ro te-label
						+--ro (technology)?

```

| | | | | ++:(generic)
| | | | | +---ro generic?
| | | | | rt-types:gener
alized-label
| | | | | +---ro direction?
| | | | | te-label-direction
| | | | | +---ro te-bandwidth
| | | | | | +---ro (technology)?
| | | | | | ++:(generic)
| | | | | | +---ro generic? te-bandwidth
| | | | | +---ro disjointness-type?
| | | | | te-types:te-path-disjointness
+---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* [node lsp-id]
        +---ro tunnel-name?
            | -> /te/lsps/lsp/tunnel-name
        +---ro node -> /te/lsps/lsp/node
        +---ro lsp-id -> /te/lsps/lsp/lsp-id
+----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
+---ro lsps
+---ro lsp* [tunnel-name lsp-id node]
+---ro tunnel-name                                string
+---ro lsp-id                                    uint16
+---ro node
|         te-types:te-node-id
+---ro source?
|         te-types:te-node-id
+---ro destination?
|         te-types:te-node-id
+---ro tunnel-id?                                uint16
+---ro extended-tunnel-id?                       yang:dotted-quad
+---ro operational-state?                         identityref
+---ro signaling-type?                           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

rpcs:
  +---x tunnels-path-compute
  |   +---w input
  |   |   +---w path-compute-info
  |   +---ro output
  |   +---ro path-compute-result
  +---x tunnels-actions
  |   +---w input
  |   |   +---w tunnel-info
  |   |   |   +---w (filter-type)
  |   |   |   |   +---:(all-tunnels)
  |   |   |   |   |   +---w all          empty
  |   |   |   |   +---:(one-tunnel)
  |   |   |   +---w tunnel?      tunnel-ref
  |   +---w action-info
  |   |   +---w action?          identityref
  |   |   +---w disruptive?     empty
  +---ro output
  |   +---ro action-result?      identityref

```

Figure 8: TE Tunnel generic model YANG tree diagram

5.3. YANG Module

The generic TE YANG module 'ietf-te' imports the following modules:

- * ietf-yang-types and ietf-inet-types defined in [\[RFC6991\]](#)
- * ietf-te-types defined in [\[RFC8776\]](#)

This module references the following documents: [\[RFC6991\]](#), [\[RFC4875\]](#), [\[RFC7551\]](#), [\[RFC4206\]](#), [\[RFC4427\]](#), [\[RFC4872\]](#), [\[RFC3945\]](#), [\[RFC3209\]](#), [\[RFC6780\]](#), [\[RFC8800\]](#), and [\[RFC7308\]](#).

Internet-Draft

TE YANG Data Model

February 2022

```
<CODE BEGINS> file "ietf-te@2021-10-22.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
      "RFC8776: Common YANG Data Types for Traffic Engineering.";
  }
  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>

     Editor:   Tarek Saad
               <mailto:tsaad@juniper.net>
```

Editor: Rakesh Gandhi
<mailto:rgandhi@cisco.com>

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

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

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

Saad, et al.

Expires 11 August 2022

[Page 61]

Internet-Draft

TE YANG Data Model

February 2022

Editor: Igor Bryskin
<mailto:i_bryskin@yahoo.com>";
description
"YANG data module for TE configuration, state, and RPCs.
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 2021-10-22 {
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 has failed because there is no topology
            with the provided topology-identifier.";
    }

    identity path-computation-error-no-dependent-server {
        base path-computation-error-reason;
        description
            "Path computation has failed because one or more dependent

```

```

        path computation servers are unavailable.
        The dependent path computation server could be
        a Backward-Recursive Path Computation (BRPC) downstream
        PCE or a child PCE.";
    reference
        "RFC5441, RFC8685";
    }

    identity path-computation-error-pce-unavailable {
        base path-computation-error-reason;
        description
            "Path computation has failed because PCE is not available.";
        reference
            "RFC5440";
    }

    identity path-computation-error-no-inclusion-hop {
        base path-computation-error-reason;
        description
            "Path computation has failed because there is no
            node or link provided by one or more inclusion hops.";
        reference

```

```

    "RFC8685";
}

identity path-computation-error-destination-unknown-in-domain {
    base path-computation-error-reason;
    description
        "Path computation has failed because the destination node is
        unknown in indicated destination domain.";
    reference
        "RFC8685";
}

identity path-computation-error-no-resource {
    base path-computation-error-reason;
    description
        "Path computation has failed because there is no
        available resource in one or more domains.";
    reference
        "RFC8685";
}

identity path-computation-error-child-pce-unresponsive {
    base path-computation-error-reason;
    description
        "Path computation has failed because child PCE is not
        responsive.";
}

```

```

    reference
        "RFC8685";
}

identity path-computation-error-destination-domain-unknown {
    base path-computation-error-reason;
    description
        "Path computation has failed because the destination domain
        was unknown.";
    reference
        "RFC8685";
}

identity path-computation-error-p2mp {
    base path-computation-error-reason;
}

```



```

    description
      "Path computation has failed because of P2MP reachability
        problem.";
    reference
      "RFC8306";
  }

  identity path-computation-error-no-gco-migration {
    base path-computation-error-reason;
    description
      "Path computation has failed because of no Global Concurrent
        Optimization (GCO) migration path found.";
    reference
      "RFC5557";
  }

  identity path-computation-error-no-gco-solution {
    base path-computation-error-reason;
    description
      "Path computation has failed because of no GCO solution
        found.";
    reference
      "RFC5557";
  }

  identity path-computation-error-path-not-found {
    base path-computation-error-reason;
    description
      "Path computation no path found error reason.";
    reference
      "RFC5440";
  }

```

```

  identity path-computation-error-pks-expansion {
    base path-computation-error-reason;
    description
      "Path computation has failed because of Path-Key Subobject
        (PKS) expansion failure.";
    reference
      "RFC5520";
  }

```

```

identity path-computation-error-brpc-chain-unavailable {
  base path-computation-error-reason;
  description
    "Path computation has failed because PCE BRPC chain
    unavailable.";
  reference
    "RFC5441";
}

identity path-computation-error-source-unknown {
  base path-computation-error-reason;
  description
    "Path computation has failed because source node is unknown.";
  reference
    "RFC5440";
}

identity path-computation-error-destination-unknown {
  base path-computation-error-reason;
  description
    "Path computation has failed because destination node is
    unknown.";
  reference
    "RFC5440";
}

identity path-computation-error-no-server {
  base path-computation-error-reason;
  description
    "Path computation has failed because path computation
    server is unavailable.";
  reference
    "RFC5440";
}

identity tunnel-actions-type {
  description
    "TE tunnel actions type.";
}

```

```

identity tunnel-action-reoptimize {

```

```

base tunnel-actions-type;
description
    "Reoptimize tunnel action type.";
}

identity tunnel-admin-auto {
    base te-types:tunnel-admin-state-type;
    description
        "Tunnel administrative auto state. The administrative status
        in state datastore transitions to 'tunnel-admin-up' when the
        tunnel used by the client layer, and to 'tunnel-admin-down'
        when it is not used by the client layer.";
}

identity association-type-diversity {
    base te-types:association-type;
    description
        "Association Type diversity used to associate LSPs whose paths
        are to be diverse from each other.";
    reference
        "RFC8800";
}

identity protocol-origin-type {
    description
        "Base identity for protocol origin type.";
}

identity protocol-origin-api {
    base protocol-origin-type;
    description
        "Protocol origin is via Application Programmable Interface
        (API).";
}

identity protocol-origin-pcep {
    base protocol-origin-type;
    description
        "Protocol origin is Path Computation Engine Protocol (PCEP).";
    reference "RFC5440";
}

identity protocol-origin-bgp {
    base protocol-origin-type;
    description
        "Protocol origin is Border Gateway Protocol (BGP).";
    reference "RFC5512";
}

typedef tunnel-ref {

```

```
type leafref {
  path "/te:te/te:tunnels/te:tunnel/te:name";
}
description
  "This type is used by data models that need to reference
  configured TE tunnel.";
}

typedef path-ref {
  type union {
    type leafref {
      path "/te:te/te:tunnels/te:tunnel/"
        + "te:primary-paths/te:primary-path/te:name";
    }
    type leafref {
      path "/te:te/te:tunnels/te:tunnel/"
        + "te:secondary-paths/te: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.";
}

typedef te-gen-node-id {
  type union {
    type te-types:te-node-id;
    type inet:ip-address;
  }
  description
    "Generic type that identifies a node in a TE topology.";
}

/**
 * TE tunnel generic groupings
 */

grouping te-generic-node-id {
  description
    "A reusable grouping for a TE generic node identifier.";
  leaf id {
    type te-gen-node-id;
    description
      "The identifier of the node. Can be represented as IP
      address or dotted quad address.";
  }
}
```

```
leaf type {  
    type enumeration {
```

```
        enum ip {  
            description  
                "IP address representation of the node identifier.";  
        }  
        enum dotted-quad {  
            description  
                "Dotted quad address representation of the node  
                identifier.";  
        }  
    }  
    description  
        "Type of node identifier representation.";  
}  
  
grouping primary-path {  
    description  
        "The tunnel primary path properties.";  
    uses path-common-properties;  
    uses path-preference;  
    uses k-requested-paths;  
    uses path-compute-info;  
    uses path-state;  
}  
  
grouping primary-reverse-path {  
    description  
        "The tunnel primary reverse path properties.";  
    reference  
        "RFC7551";  
    uses path-common-properties;  
    uses path-compute-info;  
    uses path-state;  
}  
  
grouping secondary-path {  
    description  
        "The tunnel secondary path properties.";  
    uses path-common-properties;
```

```

    uses path-preference;
    uses path-compute-info;
    uses protection-restoration-properties;
    uses path-state;
}

grouping secondary-reverse-path {
    description
        "The tunnel secondary reverse path properties.";
}

```

```

    uses path-common-properties;
    uses path-preference;
    uses path-compute-info;
    uses protection-restoration-properties;
    uses path-state;
}

grouping path-common-properties {
    description
        "Common path attributes.";
    leaf name {
        type string;
        description
            "TE path name.";
    }
    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).";
    }
    container path-computation-server {
        when "derived-from-or-self(..path-computation-method, "
            + "'te-types:path-externally-queried')";
        description
            "The path-computation server when the path is
            externally queried.";
    }
}

```

```

    uses te-generic-node-id;
    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 "derived-from-or-self(..../path-computation-method, "
        + "'te-types:path-locally-computed')";
    type boolean;

```

```

    default "true";
    description
        "When 'true' indicates the path is dynamically computed
        and/or validated against the Traffic-Engineering Database
        (TED), and when 'false' indicates no validation against
        the TED is required.";
}
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.";
}
}

/* This grouping will be re-used in path-computation rpc */

```

```

grouping path-compute-info {
  description
    "Attributes used for path computation request.";
  uses tunnel-associations-properties;
  uses te-types:generic-path-optimization;
  leaf named-path-constraint {
    if-feature "te-types:named-path-constraints";
    type leafref {
      path "/te:te/te:globals/te:named-path-constraints/"
        + "te:named-path-constraint/te:name";
    }
    description
      "Reference to a globally defined named path constraint set.";
  }
  uses path-constraints-common;
}

/* This grouping will be re-used in path-computation rpc */

grouping path-preference {
  description
    "The path preference.";
  leaf preference {

```

```

    type uint8 {
      range "1..255";
    }
    default "1";
    description
      "Specifies a preference for this path. The lower the number
        higher the preference.";
  }
}

/* This grouping will be re-used in path-computation rpc */

grouping k-requested-paths {
  description
    "The k-shortest paths requests.";
  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. The value 0 all possible paths.";
    }
}

grouping path-properties {
    description
        "TE computed path properties grouping.";
    uses te-types:generic-path-properties {
        augment "path-properties" {
            description
                "additional path properties returned by path computation.";
            uses te-types:te-bandwidth;
            leaf disjointness-type {
                type te-types:te-path-disjointness;
                config false;
                description
                    "The type of resource disjointness.
                    When reported for a primary path, it represents the
                    minimum level of disjointness of all the secondary
                    paths.
                    When reported for a secondary path, it represents the
                    disjointness of the secondary path.";
            }
        }
    }
}

```

```

grouping path-state {
    description
        "TE per path state parameters.";
    uses path-computation-response;
    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
        "The TE LSPs container.";
    list lsp {
        key "node lsp-id";
        description
            "List of LSPs associated with the tunnel.";
        leaf tunnel-name {
            type leafref {
                path "/te:te/te:lsps/te:lsp/te:tunnel-name";
            }
            description "TE tunnel name.";
        }
        leaf node {
            type leafref {
                path "/te:te/te:lsps/te:lsp/te:node";
            }
            description "The node where the LSP state resides on.";
        }
        leaf lsp-id {
            type leafref {
                path "/te:te/te:lsps/te:lsp/te:lsp-id";
            }
            description "The TE LSP identifier.";
        }
    }
}
}
}

```

/* This grouping will be re-used in path-computation rpc */

```

grouping path-computation-response {
    description
        "Attributes reported by path computation response.";
    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.";
    }
}
}
container computed-path-error-infos {
    config false;
    description
        "Path computation information container.";
    list computed-path-error-info {
        description
            "List of path computation info entries.";
        leaf error-description {
            type string;
            description
                "Textual representation of the error occurred during
                path computation.";
        }
        leaf error-timestamp {
            type yang:date-and-time;
            description
                "Timestamp of last path computation attempt.";
        }
        leaf error-reason {
            type identityref {
                base path-computation-error-reason;
            }
            description
                "Reason for the path computation error.";
        }
    }
}

```

```
    }
  }
}

grouping lsp-provisioning-error-info {
  description
    "Grouping for LSP provisioning error information.";
  container lsp-provisioning-error-infos {
    config false;
    description
      "LSP provisioning error information.";
    list lsp-provisioning-error-info {
      description
        "List of LSP provisioning error info entries.";
      leaf error-description {
        type string;
        description
          "Textual representation of the error occurred during
            path computation.";
      }
      leaf error-timestamp {
        type yang:date-and-time;
        description
          "Timestamp of when the reported error occurred.";
      }
      leaf error-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
          "Node identifier of node where error occurred.";
      }
      leaf error-link-id {
        type te-types:te-tp-id;
        default "0";
        description
          "Link ID where the error occurred.";
      }
    }
  }
}

grouping 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
        has not been committed at the data plane.";
    }
  }
}
default "working";
description
  "LSP role type.";
reference
```

```

    "RFC4872, section 4.2.1";
}
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 tunnel-associations-properties {
  description
    "TE tunnel association grouping.";
  container association-objects {
    description
      "TE tunnel associations.";
    list association-object {
      key "association-key";
      unique "type id source/id source/type";
      description
        "List of association base objects.";
      reference
        "RFC4872";
      leaf association-key {
        type string;
        description
          "Association key used to identify a specific
            association in the list";
      }
      leaf type {
        type identityref {
          base te-types:association-type;
        }
        description
          "Association type.";
        reference
          "RFC4872";
      }
      leaf id {
        type uint16;
        description
          "Association identifier.";
        reference
          "RFC4872";
      }
    }
  }
}

```

```

}
container source {
  uses te-generic-node-id;
  description
    "Association source.";
}

```

```

        reference
            "RFC4872";
    }
}
list association-object-extended {
    key "association-key";
    unique
        "type id source/id source/type global-source extended-id";
    description
        "List of extended association objects.";
    reference
        "RFC6780";
    leaf association-key {
        type string;
        description
            "Association key used to identify a specific
            association in the list";
    }
    leaf type {
        type identityref {
            base te-types:association-type;
        }
        description
            "Association type.";
        reference
            "RFC4872, RFC6780";
    }
    leaf id {
        type uint16;
        description
            "Association identifier.";
        reference
            "RFC4872, RFC6780";
    }
    container source {
        uses te-generic-node-id;
        description
            "Association source.";
        reference
            "RFC4872, RFC6780";
    }
    leaf global-source {
        type uint32;
    }
}

```

```

        description
            "Association global source.";
        reference
            "RFC6780";
    }
    leaf extended-id {
        type yang:hex-string;
        description
            "Association extended identifier.";
        reference
            "RFC6780";
    }
}
}
}

/* TE tunnel configuration/state grouping */
/* These grouping will be re-used in path-computation rpc */

grouping encoding-and-switching-type {
    description
        "Common grouping to define the LSP encoding and
        switching types";
    leaf encoding {
        type identityref {
            base te-types:lsp-encoding-types;
        }
        description
            "LSP encoding type.";
        reference
            "RFC3945";
    }
    leaf switching-type {
        type identityref {
            base te-types:switching-capabilities;
        }
        description
            "LSP switching type.";
        reference
            "RFC3945";
    }
}

grouping tunnel-common-attributes {
    description
        "Common grouping to define the TE tunnel parameters";
    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 identifier.";
    }
    leaf src-tunnel-tp-id {
        type binary;
        description
            "TE tunnel source termination point identifier.";
    }
    leaf dst-tunnel-tp-id {
        type binary;
        description
            "TE tunnel destination termination point identifier.";
    }
    leaf bidirectional {
        type boolean;
        default "false";
        description
            "Indicates a bidirectional co-routed LSP.";
    }
}

grouping tunnel-hierarchy-properties {
    description
        "A grouping for TE tunnel hierarchy information.";
    container hierarchy {
        description
            "Container for TE hierarchy related information.";
        container dependency-tunnels {
            description
                "List of tunnels that this tunnel can be potentially
                dependent on.";
            list dependency-tunnel {
                key "name";
                description
                    "A tunnel entry that this tunnel can potentially depend
                    on.";
                leaf name {
```

```

    type leafref {
      path "/te:te/te:tunnels/te:tunnel/te:name";
      require-instance false;
    }
    description
      "Dependency tunnel name. The tunnel may not have been

```

```

      instantiated yet.";
    }
    uses encoding-and-switching-type;
  }
}
container hierarchical-link {
  description
    "Identifies a hierarchical link (in client layer)
    that this tunnel is associated with.";
  reference
    "RFC4206";
  leaf local-te-node-id {
    type te-types:te-node-id;
    default "0.0.0.0";
    description
      "The local TE node identifier.";
  }
  leaf local-te-link-tp-id {
    type te-types:te-tp-id;
    default "0";
    description
      "The 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 {
    description
      "The topology identifier where the hierarchical link
      supported by this TE tunnel is instantiated.";
  }
}
}

```

```

    }
}

grouping tunnel-properties {
    description
        "Top level grouping for tunnel properties.";
    leaf name {
        type string;
        description
            "TE tunnel name.";
    }
    leaf alias {
        type string;
    }
}

```

```

        description
            "An alternate name of the TE tunnel that can be modified
            anytime during its lifetime.";
    }
    leaf identifier {
        type uint32;
        description
            "TE tunnel Identifier.";
        reference
            "RFC3209";
    }
    leaf color {
        type uint32;
        description "The color associated with the TE tunnel.";
        reference "RFC9012";
    }
    leaf description {
        type string;
        default "None";
        description
            "Textual description for this TE tunnel.";
    }
    leaf admin-state {
        type identityref {
            base te-types:tunnel-admin-state-type;
        }
        default "te-types:tunnel-admin-state-up";
        description

```

```

        "TE tunnel administrative state.";
    }
    leaf operational-state {
        type identityref {
            base te-types:tunnel-state-type;
        }
        config false;
        description
            "TE tunnel operational state.";
    }
    uses encoding-and-switching-type;
    uses tunnel-common-attributes;
    container controller {
        description
            "Contains tunnel data relevant to external controller(s).
            This target node may be augmented by external module(s),
            for example, to add data for PCEP initiated and/or
            delegated tunnels.";
        leaf protocol-origin {
            type identityref {

```

```

        base protocol-origin-type;
    }
    description
        "The protocol origin for instantiating the tunnel.";
    }
    leaf controller-entity-id {
        type string;
        description
            "An identifier unique within the scope of visibility that
            associated with the entity that controls the tunnel";
        reference "RFC8232";
    }
    }
    leaf reoptimize-timer {
        type uint16;
        units "seconds";
        description
            "Frequency of reoptimization of a traffic engineered LSP.";
    }
    uses tunnel-associations-properties;
    uses protection-restoration-properties;

```



```

uses te-types:tunnel-constraints;
uses tunnel-hierarchy-properties;
container primary-paths {
  description
    "The set of primary paths.";
  list primary-path {
    key "name";
    description
      "List of primary paths for this tunnel.";
    uses primary-path;
    container primary-reverse-path {
      description
        "The reverse primary path properties.";
      uses primary-reverse-path;
      container candidate-secondary-reverse-paths {
        description
          "The set of referenced candidate reverse secondary
            paths from the full set of secondary reverse paths
            which may be used for this primary path.";
        list candidate-secondary-reverse-path {
          key "secondary-path";
          ordered-by user;
          description
            "List of candidate secondary reverse path(s)";
          leaf secondary-path {
            type leafref {
              path "../..../..../..../.."
            }
          }
        }
      }
    }
  }
}

```

```

      + "te:secondary-reverse-paths/"
      + "te:secondary-reverse-path/te:name";
    }
    description
      "A reference to the secondary reverse path that
        should be utilised when the containing primary
        reverse path option is in use.";
  }
}
}
}
}
container candidate-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-secondary-path {
  key "secondary-path";
  ordered-by user;
  description
    "List of candidate secondary paths for this tunnel.";
  leaf secondary-path {
    type leafref {
      path "../..../te:secondary-paths/"
        + "te:secondary-path/te:name";
    }
    description
      "A reference to the secondary path that should be
        utilised when the containing primary path option is
        in use.";
  }
  leaf active {
    type boolean;
    config false;
    description
      "Indicates the current active path option that has
        been selected of the candidate secondary paths.";
  }
}
}
}

```

```

}
container secondary-paths {
  description
    "The set of secondary paths.";
  list secondary-path {
    key "name";
    description

```

```

        "List of secondary paths for this tunnel.";
        uses secondary-path;
    }
}
container secondary-reverse-paths {
    description
        "The set of secondary reverse paths.";
    list secondary-reverse-path {
        key "name";
        description
            "List of secondary paths for this tunnel.";
        uses secondary-reverse-path;
    }
}
}

grouping tunnel-actions {
    description
        "Tunnel actions.";
    action tunnel-action {
        description
            "Tunnel action.";
        input {
            leaf action-type {
                type identityref {
                    base tunnel-actions-type;
                }
                description
                    "Tunnel action type.";
            }
        }
        output {
            leaf action-result {
                type identityref {
                    base te-types:te-action-result;
                }
                description
                    "The result of the tunnel action 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.";
}
}
}
}

/** End of TE tunnel groupings */
/**
 * LSP related generic groupings
 */

grouping lsp-record-route-information-state {
    description
        "LSP 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-grouping {
    description
        "LSPs state operational data grouping.";
    container lsps {
        config false;
        description
            "TE LSPs state container.";
        list lsp {
            key "tunnel-name lsp-id node";
            unique "source destination tunnel-id lsp-id "
                + "extended-tunnel-id";
            description
                "List of LSPs associated with the tunnel.";
            leaf tunnel-name {
                type string;
                description "The TE tunnel name.";
            }
            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 node {
                type te-types:te-node-id;
                description
                    "The node where the TE LSP state resides on.";
            }
            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";
    }
}

grouping named-admin-groups {
    description
        "Global named administrative groups configuration
        grouping.";
    container named-admin-groups {
        description
            "TE named admin groups container.";
        list named-admin-group {
            if-feature "te-types:extended-admin-groups";
            if-feature "te-types:named-extended-admin-groups";
            key "name";
            description
                "List of named TE admin-groups.";
        }
    }
}

```

```

        uses named-admin-groups-properties;
    }
}

/* Global named admin-srlgs configuration data */

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.";
            leaf name {

```

```

        type string;
        description
            "A string name that uniquely identifies a TE
            interface named SRLG.";
    }
    leaf value {
        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.";
    }
}

/* Global named paths constraints configuration data */

```



```

grouping path-constraints-common {
  description
    "Global named path constraints configuration
    grouping.";
  uses te-types:common-path-constraints-attributes {
    description
      "The constraints applicable to the path. This includes:
      - The path bandwidth constraint
      - The path link protection type constraint
      - The path setup/hold priority constraint
      - path signaling type constraint
      - path metric bounds constraint. The unit of path metric
        bound is interpreted in the context of the metric-type.
        For example for metric-type 'path-metric-loss', the bound
        is multiples of the basic unit 0.000003% as described
        in RFC7471 for OSPF, and RFC8570 for ISIS.
      - path affinity constraints
      - path SRLG constraints";
  }
  uses te-types:generic-path-disjointness;
  uses te-types:path-constraints-route-objects;
  container path-in-segment {
    presence "The end-to-end tunnel starts in a previous domain;
    this tunnel is a segment in the current domain.";
    description

```

```

    "If an end-to-end tunnel crosses multiple domains using
    the same technology, some additional constraints have to be
    taken in consideration in each domain.
    This TE tunnel segment is stitched to the upstream TE tunnel
    segment.";
    uses te-types:label-set-info;
  }
  container path-out-segment {
    presence
      "The end-to-end tunnel is not terminated in this domain;
      this tunnel is a segment in the current domain.";
    description
      "If an end-to-end tunnel crosses multiple domains using
      the same technology, some additional constraints have to be
      taken in consideration in each domain.

```

```

        This TE tunnel segment is stitched to the downstream TE
        tunnel segment.";
    uses te-types:label-set-info;
}
}

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
        "The list of TE tunnels.";
      uses tunnel-properties;
      uses tunnel-actions;
      uses tunnel-protection-actions;
    }
  }
}

/* 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
      "The tunnel endpoint address extracted from SESSION object.";
    reference
      "RFC3209";
  }
}

```

}

```

leaf tunnel-id {
    type uint16;
    description
        "The tunnel identifier used in the SESSION that remains
        constant over the life of the tunnel.";
    reference
        "RFC3209";
}
leaf extended-tunnel-id {
    type yang:dotted-quad;
    description
        "The LSP Extended Tunnel ID.";
    reference
        "RFC3209";
}
leaf operational-state {
    type identityref {
        base te-types:lsp-state-type;
    }
    description
        "The LSP operational state.";
}
leaf signaling-type {
    type identityref {
        base te-types:path-signaling-type;
    }
    description
        "The signaling protocol used to set up this LSP.";
}
leaf origin-type {
    type enumeration {
        enum ingress {
            description
                "Origin ingress.";
        }
        enum egress {
            description
                "Origin egress.";
        }
        enum transit {
            description
                "Origin transit.";
        }
    }
    default "ingress";
    description
        "The origin of the 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 state.";
        reference
            "RFC4872, section 4.2.1";
    }
    uses protection-restoration-properties-state;
}

/**** End of TE global groupings ****/
/**
 * TE container
 */

container te {
    presence "Enable TE feature.";
    description
        "TE global container.";
    /* TE Global Data */
    uses globals-grouping;

    /* TE Tunnel Data */
    uses tunnels-grouping;

    /* TE LSPs Data */
    uses lsps-grouping;
}

/* TE Tunnel RPCs/execution Data */
```

rpc tunnels-path-compute {

```
description
  "TE tunnels RPC nodes.";
input {
  container path-compute-info {
    /*
     * An external path compute module may augment this
     * target.
     */
    description
      "RPC input information.";
  }
}
output {
  container path-compute-result {
    /*
     * An external path compute module may augment this
     * target.
     */
    description
      "RPC output information.";
  }
}
}

rpc tunnels-actions {
  description
    "TE tunnels actions RPC";
  input {
    container tunnel-info {
      description
        "TE tunnel information.";
      choice filter-type {
        mandatory true;
        description
          "Filter choice.";
        case all-tunnels {
          leaf all {
            type empty;
            mandatory true;
          }
        }
      }
    }
  }
}
```

```

        description
            "Apply action on all TE tunnels.";
    }
}
case one-tunnel {
    leaf tunnel {
        type tunnel-ref;
        description
            "Apply action on the specific TE tunnel.";
    }
}

```

```

    }
}
}
}
container action-info {
    description
        "TE tunnel action information.";
    leaf action {
        type identityref {
            base tunnel-actions-type;
        }
        description
            "The action type.";
    }
    leaf disruptive {
        when "derived-from-or-self(..../action, "
            + "'te:tunnel-action-reoptimize')";
        type empty;
        description
            "Specifies whether or not the reoptimization action
            is allowed to be disruptive.";
    }
}
}
output {
    leaf action-result {
        type identityref {
            base te-types:te-action-result;
        }
        description
            "The result of the tunnel action operation.";
    }
}

```

```

    }
  }
}
<CODE ENDS>

```

Figure 9: TE Tunnel data model YANG module

6. TE Device YANG Model

The device TE YANG module ('ietf-te-device') models data that is specific to managing a TE device. This module augments the generic TE YANG module.

6.1. Module Structure

6.1.1. TE Interfaces

This branch of the model manages TE interfaces that are present on a device. 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)

The derived state associated with interfaces is grouped under the

interface "state" sub-container as shown in Figure 10. 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

```
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 10: TE interface state YANG subtree

[6.2.](#) Tree Diagram

Figure 11 shows the tree diagram of the device TE YANG model defined in modules 'ietf-te.yang'.

```
module: ietf-te-device
```

```

augment /te:te:
  +--rw interfaces
  |   +--rw threshold-type?          enumeration
  |   +--rw delta-percentage?        rt-types:percentage
  |   +--rw threshold-specification? enumeration
  |   +--rw up-thresholds*           rt-types:percentage
  |   +--rw down-thresholds*         rt-types:percentage
  |   +--rw up-down-thresholds*      rt-types:percentage
  |   +--rw interface* [interface]
  |       +--rw interface            if:interface-ref
  |       +--rw te-metric?
  |           |   te-types:te-metric
  |       +--rw (admin-group-type)?
  |           |   +--:(value-admin-groups)
  |           |       |   +--rw (value-admin-group-type)?
  |           |       |       |   +--:(admin-groups)
  |           |       |       |       |   +--rw admin-group?
  |           |       |       |       |       |   te-types:admin-group
  |           |       |       |       |       +--:(extended-admin-groups)
  |           |       |       |       |       {te-types:extended-admin-groups}?
  |           |       |       |       +--rw extended-admin-group?
  |           |       |       |           |   te-types:extended-admin-group
  |           |       |       +--:(named-admin-groups)
  |           |       +--rw named-admin-groups* [named-admin-group]
  |           |           {te-types:extended-admin-groups,te-types:named-
extended-admin-groups}?
  |           |       +--rw named-admin-group    leafref
  |       +--rw (srlg-type)?
  |           |   +--:(value-srlgs)
  |           |       |   +--rw values* [value]

```

```

  |       |   +--rw value    uint32
  |       +--:(named-srlgs)
  |           +--rw named-srlgs* [named-srlg]
  |               {te-types:named-srlg-groups}?
  |           +--rw named-srlg    leafref
  +--rw threshold-type?          enumeration
  +--rw delta-percentage?
  |   rt-types:percentage
  +--rw threshold-specification? enumeration
  +--rw up-thresholds*
  |   rt-types:percentage

```

```

|         +---rw down-thresholds*
|         |         rt-types:percentage
|         +---rw up-down-thresholds*
|         |         rt-types:percentage
|         +---rw switching-capabilities* [switching-capability]
|         |         +---rw switching-capability      identityref
|         |         +---rw encoding?                  identityref
|         +---ro state
|         |         +---ro te-advertisements-state
|         |         |         +---ro flood-interval?          uint32
|         |         |         +---ro last-flooded-time?       uint32
|         |         |         +---ro next-flooded-time?       uint32
|         |         |         +---ro last-flooded-trigger?    enumeration
|         |         |         +---ro advertised-level-areas* [level-area]
|         |         |         +---ro level-area      uint32
|         +---rw performance-thresholds
augment /te:te/te:globals:
+---rw lsp-install-interval?      uint32
+---rw lsp-cleanup-interval?      uint32
+---rw lsp-invalidation-interval? uint32
augment /te:te/te:tunnels/te:tunnel:
+---rw path-invalidation-action?  identityref
+---rw lsp-install-interval?      uint32
+---rw lsp-cleanup-interval?      uint32
+---rw lsp-invalidation-interval? uint32
augment /te:te/te:lsps/te:lsp:
+---ro lsp-timers
| +---ro life-time?      uint32
| +---ro time-to-install? uint32
| +---ro time-to-destroy? uint32
+---ro downstream-info
| +---ro nhop?          te-types:te-tp-id
| +---ro outgoing-interface? if:interface-ref
| +---ro neighbor
| | +---ro id?          te-gen-node-id
| | +---ro type?        enumeration
| +---ro label?         rt-types:generalized-label

```

```

+---ro upstream-info
+---ro phop?      te-types:te-tp-id
+---ro neighbor
| +---ro id?      te-gen-node-id

```

```

| +--ro type?    enumeration
+--ro label?     rt-types:generalized-label

rpcs:
  +---x link-state-update
    +---w input
      +---w (filter-type)
        +--:(match-all)
          | +---w all          empty
          +--:(match-one-interface)
            +---w interface?  if:interface-ref

```

Figure 11: TE Tunnel device model YANG tree diagram

6.3. YANG Module

The device TE 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 [[RFC8776](#)]
- * ietf-te defined in this document

```

<CODE BEGINS> file "ietf-te-device@2021-10-22.yang"
module ietf-te-device {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-device";

  /* Replace with IANA when assigned */

  prefix te-dev;

  /* Import TE module */

  import ietf-te {
    prefix te;
    reference
      "draft-ietf-teas-yang-te: A YANG Data Model for Traffic

```

```
    Engineering Tunnels and Interfaces";
}

/* Import TE types */

import ietf-te-types {
    prefix te-types;
    reference
        "RFC8776: Common YANG Data Types for Traffic Engineering.";
}
import ietf-interfaces {
    prefix if;
    reference
        "RFC8343: A YANG Data Model for Interface Management";
}
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>

    Editor:     Tarek Saad
                <mailto:tsaad@juniper.net>

    Editor:     Rakesh Gandhi
                <mailto:rgandhi@cisco.com>

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

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

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

    Editor:     Igor Bryskin
                <mailto:i_bryskin@yahoo.com>";
description
    "YANG data module for TE device configurations,
```

state, and RPCs. 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 2021-10-22 {
  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-info {
  description
    "TE LSP device state grouping.";
  container lsp-timers {
    when "../te:origin-type = 'ingress'" {
      description
        "Applicable to ingress LSPs only.";
    }
  }
  description
```

```

    "Ingress LSP timers.";
leaf life-time {
    type uint32;
    units "seconds";
    description
        "TE LSP lifetime.";
}
leaf time-to-install {

```

```

    type uint32;
    units "seconds";
    description
        "TE LSP installation delay time.";
}
leaf time-to-destroy {
    type uint32;
    units "seconds";
    description
        "TE LSP expiration delay time.";
}
}
container downstream-info {
    when "../te:origin-type != 'egress'" {
        description
            "Downstream information of the LSP.";
    }
    description
        "downstream information.";
    leaf nhop {
        type te-types:te-tp-id;
        description
            "downstream next-hop address.";
    }
    leaf outgoing-interface {
        type if:interface-ref;
        description
            "downstream interface.";
    }
    container neighbor {
        uses te:te-generic-node-id;
        description
            "downstream neighbor address.";
    }
}

```

```

    }
    leaf label {
        type rt-types:generalized-label;
        description
            "downstream label.";
    }
}
container upstream-info {
    when "../te:origin-type != 'ingress'" {
        description
            "Upstream information of the LSP.";
    }
    description
        "upstream information.";
    leaf phop {

```

```

        type te-types:te-tp-id;
        description
            "upstream next-hop or previous-hop address.";
    }
    container neighbor {
        uses te:te-generic-node-id;
        description
            "upstream neighbor address.";
    }
    leaf label {
        type rt-types:generalized-label;
        description
            "upstream label.";
    }
}
}

/**
 * Device general groupings.
 */

grouping lsp-device-timers {
    description
        "Device TE LSP timers configs.";
    leaf lsp-install-interval {
        type uint32;

```



```

        units "seconds";
        description
            "TE LSP installation delay time.";
    }
    leaf lsp-cleanup-interval {
        type uint32;
        units "seconds";
        description
            "TE LSP cleanup delay time.";
    }
    leaf lsp-invalidation-interval {
        type uint32;
        units "seconds";
        description
            "TE LSP path invalidation before taking action delay time.";
    }
}

/**
 * TE global device groupings
 */
/* TE interface container data */

```

```

grouping interfaces-grouping {
    description
        "TE interface 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 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";
}
description
  "Value of the SRLG";

```

```

    }
  }
}
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.";
        }
    }
}

```

Internet-Draft

TE YANG Data Model

February 2022

```
    }
  }
  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
    "TE interface metric grouping.";
  leaf te-metric {
    type te-types:te-metric;
    description
      "TE interface 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 advertised-level-areas {
    key "level-area";
    description
        "List of level-areas that the TE interface is advertised

```

```

        in.";
    leaf level-area {
        type uint32;
        description
            "The IGP area or level where the TE interface link 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;
}

/* TE tunnels device configuration augmentation */

augment "/te:te/te:tunnels/te:tunnel" {
  description
    "Tunnel device dependent augmentation.";
  leaf path-invalidation-action {
    type identityref {
      base te-types:path-invalidation-action-type;
    }
    description
      "Tunnel path invalidation action.";
  }
  uses lsp-device-timers;
}

```

```

/* TE LSPs device state augmentation */

augment "/te:te/te:lsps/te:lsp" {
  description
    "TE LSP device dependent augmentation.";
  uses lsps-device-info;
}

```

```

/* TE interfaces RPCs/execution Data */

rpc link-state-update {
  description
    "Triggers a link state update for the specific interface.";
  input {
    choice filter-type {
      mandatory true;
      description
        "Filter choice.";
      case match-all {
        leaf all {
          type empty;
          mandatory true;
          description
            "Match all TE interfaces.";
        }
      }
      case match-one-interface {
        leaf interface {
          type if:interface-ref;
          description
            "Match a specific TE interface.";
        }
      }
    }
  }
}
}
<CODE ENDS>

```

Figure 12: TE device data model YANG module

7. Notifications

Notifications are a key component of any topology data model.

[RFC8639] and [[RFC8641](#)] define a subscription mechanism and a push mechanism for YANG datastores. These mechanisms currently allow the user to:

- * Subscribe to notifications on a per-client basis.
- * Specify subtree filters or XML Path Language (XPath) filters so that only contents of interest will be sent.
- * Specify either periodic or on-demand notifications.

8. TE Generic and Helper YANG Modules

9. 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.

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

URI: urn:ietf:params:xml:ns:yang:ietf-te-device
 Registrant Contact: The IESG.
 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
 Namespace: urn:ietf:params:xml:ns:yang:ietf-te
 Prefix: te
 Reference: RFCXXXX

Name: ietf-te-device
 Namespace: urn:ietf:params:xml:ns:yang:ietf-te-device
 Prefix: te-device
 Reference: RFCXXXX

10. 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:

`"/te/globals"`: This module specifies the global TE configurations on a device. Unauthorized access to this container could cause the device to ignore packets it should receive and process.

`"/te/tunnels"`: This list specifies the configuration and state of TE Tunnels present on the device or controller. Unauthorized access to this list could cause the device to ignore packets it should receive and process. An attacker may also use state to derive information about the network topology, and subsequently orchestrate further attacks.

`"/te/interfaces"`: This list specifies the configuration and state TE interfaces on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

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:

`"/te/lsp"`: this list contains information state about established LSPs in the network. An attacker can use this information to derive information about the network topology, and subsequently orchestrate further attacks.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

"/te/tunnels-actions": using this RPC, an attacker can modify existing paths that may be carrying live traffic, and hence result to interruption to services carried over the network.

"/te/tunnels-path-compute": using this RPC, an attacker can retrieve secured information about the network provider which can be used to orchestrate further attacks.

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

[11.](#) 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 like to thank Tom Petch for reviewing and providing useful feedback about the document. The authors would also like to thank Loa Andersson, Lou Berger, Sergio Belotti, Italo Busi, Carlo Perocchio, Francesco Lazzeri, Aihua Guo, Dhruv Dhody, and Raqib Jones for providing useful feedback on this document.

[12.](#) Contributors

Himanshu Shah
Ciena

Email: hshah@ciena.com

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

13. [Appendix A](#): Data Tree Examples

This section contains examples of use of the model with RESTCONF [[RFC8040](#)] and JSON encoding.

For the example we will use a 4 node MPLS network where RSVP-TE MPLS Tunnels can be setup. The loopbacks of each router are shown. The network in Figure 13 will be used in the examples described in the following sections.

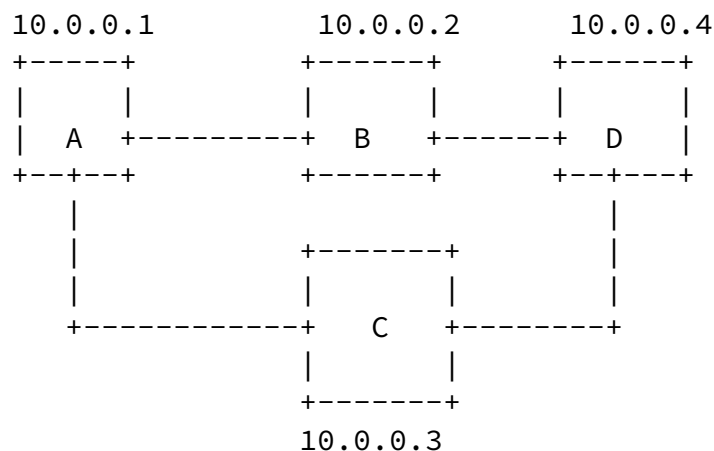


Figure 13: TE network used in data tree examples

13.1. [Basic Tunnel Setup](#)

This example uses the TE Tunnel YANG data model defined in this document to create an RSVP-TE signaled Tunnel of packet LSP encoding type. First, the TE Tunnel is created with no specific restrictions or constraints (e.g., protection or restoration). The TE Tunnel ingresses on router A and egresses on router D.

In this case, the TE Tunnel is created without specifying additional information about the primary paths.

```
POST /restconf/data/ietf-te:te/tunnels HTTP/1.1
Host: example.com
Accept: application/yang-data+json
```


Content-Type: application/yang-data+json

```
{
  "ietf-te:tunnel": [
    {
      "name": "Example_LSP_Tunnel_A_2",
      "encoding": "te-types:lsp-encoding-packet",
      "admin-state": "te-types:tunnel-state-up",
      "source": "10.0.0.1",
      "destination": "10.0.0.4",
      "bidirectional": "false",
      "signaling-type": "te-types:path-setup-rsvp"
    }
  ]
}
```

Saad, et al.

Expires 11 August 2022

[Page 120]

Internet-Draft

TE YANG Data Model

February 2022

[13.2.](#) Global Named Path Constraints

This example uses the YANG data model to create a 'named path constraint' that can be reference by TE Tunnels. The path constraint, in this case, limits the TE Tunnel hops for the computed path.

```
POST /restconf/data/ietf-te:te/globals/named-path-constraints HTTP/1.1
Host: example.com
Accept: application/yang-data+json
Content-Type: application/yang-data+json
```

```
{
  "ietf-te:named-path-constraint": {
    "name": "max-hop-3",
    "path-metric-bounds": {
      "path-metric-bound": {
        "metric-type": "te-types:path-metric-hop",
        "upper-bound": "3"
      }
    }
  }
}
```

```
}
```

[13.3.](#) Tunnel with Global Path Constraint

In this example, the previously created 'named path constraint' is applied to the TE Tunnel created in [Section 13.1](#).

```
POST /restconf/data/ietf-te:te/tunnels HTTP/1.1
```

```
Host: example.com
```

```
Accept: application/yang-data+json
```

```
Content-Type: application/yang-data+json
```

```
{
```

```
  "ietf-te:ietf-tunnel": [
```

```
    {
```

```
      "name": "Example_LSP_Tunnel_A_4_1",
```

```
      "encoding": "te-types:lsp-encoding-packet",
```

```
      "description": "Simple_LSP_with_named_path",
```

```
      "admin-state": "te-types:tunnel-state-up",
```

```
      "source": "10.0.0.1",
```

```
      "destination": "10.0.0.4",
```

```
      "signaling-type": "path-setup-rsvp",
```

```

    "bidirectional": "false",
    "primary-paths": [
      {
        "primary-path": {
          "name": "Simple_LSP_1",
          "use-path-computation": "true",
          "named-path-constraint": "max-hop-3"
        }
      }
    ]
  }
}

```

[13.4.](#) Tunnel with Per-tunnel Path Constraint

In this example, the a per tunnel path constraint is explicitly indicated under the TE Tunnel created in [Section 13.1](#) to constrain the computed path for the tunnel.

```

POST /restconf/data/ietf-te:te/tunnels HTTP/1.1
Host: example.com
Accept: application/yang-data+json
Content-Type: application/yang-data+json

{
  "ietf-te:tunnel": [
    {

```

```

"name": "Example_LSP_Tunnel_A_4_2",
"encoding": "te-types:lsp-encoding-packet",
"admin-state": "te-types:tunnel-state-up",
"source": "10.0.0.1",
"destination": "10.0.0.4",
"bidirectional": "false",
"signaling-type": "te-types:path-setup-rsvp",
"primary-paths": {
  "primary-path": [
    {
      "name": "path1",
      "path-metric-bounds": {
        "path-metric-bound": [
          {
            "metric-type": "te-types:path-metric-hop",
            "upper-bound": "3"
          }
        ]
      }
    }
  ]
}
}

```

[13.5.](#) Tunnel State

In this example, the 'GET' query is sent to return the state stored about the tunnel.

```

GET /restconf/data/ietf-te:te/tunnels/tunnel="Example_LSP_Tunnel_A_4_1"
    /p2p-primary-paths/ HTTP/1.1
Host: example.com
Accept: application/yang-data+json

```

The request, with status code 200 would include, for example, the following json:

```

"ietf-te:primary-paths": {
  "primary-path": [
    {
      "name": "path1",
      "path-computation-method": "te-types:path-locally-computed",
      "computed-paths-properties": {
        "computed-path-properties": [
          {
            "k-index": "1",
            "path-properties": {
              "path-route-objects": {
                "path-route-object": [
                  {
                    "index": "1",
                    "numbered-node-hop": {
                      "node-id": "10.0.0.2"
                    }
                  },
                  {
                    "index": "2",
                    "numbered-node-hop": {
                      "node-id": "10.0.0.4"
                    }
                  }
                ]
              }
            }
          }
        ]
      },
      "lsp": {
        "lsp": [
          {
            "tunnel-name": "Example_LSP_Tunnel_A_4_1",
            "node": "10.0.0.1 ",
            "lsp-id": "25356"
          }
        ]
      }
    }
  ]
}

```

[14.](#) References

[14.1.](#) Normative References

-
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", [RFC 3209](#), DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions", [RFC 3473](#), DOI 10.17487/RFC3473, January 2003, <<https://www.rfc-editor.org/info/rfc3473>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC3945] Mannie, E., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", [RFC 3945](#), DOI 10.17487/RFC3945, October 2004, <<https://www.rfc-editor.org/info/rfc3945>>.
- [RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", [RFC 4206](#), DOI 10.17487/RFC4206, October 2005, <<https://www.rfc-editor.org/info/rfc4206>>.
- [RFC4872] Lang, J.P., Ed., Rekhter, Y., Ed., and D. Papadimitriou, Ed., "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery", [RFC 4872](#), DOI 10.17487/RFC4872, May 2007, <<https://www.rfc-editor.org/info/rfc4872>>.
- [RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", [RFC 4875](#), DOI 10.17487/RFC4875, May 2007, <<https://www.rfc-editor.org/info/rfc4875>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#),

-
- [RFC6107] Shiomoto, K., Ed. and A. Farrel, Ed., "Procedures for Dynamically Signaled Hierarchical Label Switched Paths", [RFC 6107](#), DOI 10.17487/RFC6107, February 2011, <<https://www.rfc-editor.org/info/rfc6107>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", [RFC 6241](#), DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", [RFC 6242](#), DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6780] Berger, L., Le Faucheur, F., and A. Narayanan, "RSVP ASSOCIATION Object Extensions", [RFC 6780](#), DOI 10.17487/RFC6780, October 2012, <<https://www.rfc-editor.org/info/rfc6780>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", [RFC 6991](#), DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", [RFC 7308](#), DOI 10.17487/RFC7308, July 2014, <<https://www.rfc-editor.org/info/rfc7308>>.
- [RFC7551] Zhang, F., Ed., Jing, R., and R. Gandhi, Ed., "RSVP-TE Extensions for Associated Bidirectional Label Switched Paths (LSPs)", [RFC 7551](#), DOI 10.17487/RFC7551, May 2015, <<https://www.rfc-editor.org/info/rfc7551>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", [RFC 7950](#), DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", [RFC 8040](#), DOI 10.17487/RFC8040, January 2017,

<<https://www.rfc-editor.org/info/rfc8040>>.

- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

Saad, et al.

Expires 11 August 2022

[Page 126]

Internet-Draft

TE YANG Data Model

February 2022

- [RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", [RFC 8294](#), DOI 10.17487/RFC8294, December 2017, <<https://www.rfc-editor.org/info/rfc8294>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", [BCP 215](#), [RFC 8340](#), DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, [RFC 8341](#), DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", [RFC 8342](#), DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", [RFC 8343](#), DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", [RFC 8446](#), DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", [RFC 8639](#), DOI 10.17487/RFC8639, September 2019, <<https://www.rfc-editor.org/info/rfc8639>>.

- [RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", [RFC 8641](#), DOI 10.17487/RFC8641, September 2019, <<https://www.rfc-editor.org/info/rfc8641>>.
- [RFC8776] Saad, T., Gandhi, R., Liu, X., Beeram, V., and I. Bryskin, "Common YANG Data Types for Traffic Engineering", [RFC 8776](#), DOI 10.17487/RFC8776, June 2020, <<https://www.rfc-editor.org/info/rfc8776>>.
- [RFC8795] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Gonzalez de Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", [RFC 8795](#), DOI 10.17487/RFC8795, August 2020, <<https://www.rfc-editor.org/info/rfc8795>>.

[14.2](#). Informative References

Saad, et al. Expires 11 August 2022 [Page 127]

Internet-Draft TE YANG Data Model February 2022

- [I-D.ietf-spring-segment-routing-policy] Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, [draft-ietf-spring-segment-routing-policy-16](#), 28 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-16.txt>>.
- [I-D.ietf-teas-yang-rsvp] Beeram, V. P., Saad, T., Gandhi, R., Liu, X., and I. Bryskin, "A YANG Data Model for Resource Reservation Protocol (RSVP)", Work in Progress, Internet-Draft, [draft-ietf-teas-yang-rsvp-17](#), 9 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-teas-yang-rsvp-17.txt>>.
- [RFC4427] Mannie, E., Ed. and D. Papadimitriou, Ed., "Recovery (Protection and Restoration) Terminology for Generalized Multi-Protocol Label Switching (GMPLS)", [RFC 4427](#), DOI 10.17487/RFC4427, March 2006, <<https://www.rfc-editor.org/info/rfc4427>>.
- [RFC8800] Litkowski, S., Sivabalan, S., Barth, C., and M. Negi, "Path Computation Element Communication Protocol (PCEP) Extension for Label Switched Path (LSP) Diversity

Constraint Signaling", [RFC 8800](https://www.rfc-editor.org/info/rfc8800), DOI 10.17487/RFC8800, July 2020, <<https://www.rfc-editor.org/info/rfc8800>>.

[RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", [RFC 9012](https://www.rfc-editor.org/info/rfc9012), DOI 10.17487/RFC9012, April 2021, <<https://www.rfc-editor.org/info/rfc9012>>.

Authors' Addresses

Tarek Saad
Juniper Networks

Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Saad, et al.

Expires 11 August 2022

[Page 128]

Internet-Draft

TE YANG Data Model

February 2022

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Igor Bryskin
Individual

Email: i_bryskin@yahoo.com

Oscar Gonzalez de Dios
Telefonica

Email: oscar.gonzalezdedios@telefonica.com