

Network Working Group  
Internet Draft  
Intended status: Standards Track

Chandra Ramachandran  
Juniper Networks  
Ina Minei  
Google, Inc  
Dante Pacella  
Verizon  
Tarek Saad  
Cisco Systems Inc.

Expires: November 7, 2016

May 7, 2016

Refresh Interval Independent FRR Facility Protection  
draft-chandra-mpls-ri-rsvp-frr-04

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

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

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

This Internet-Draft will expire on November 7, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Abstract

RSVP-TE relies on periodic refresh of RSVP messages to synchronize and maintain the LSP related states along the reserved path. In the absence of refresh messages, the LSP related states are automatically deleted. Reliance on periodic refreshes and refresh timeouts are problematic from the scalability point of view. The number of RSVP-TE LSPs that a router needs to maintain has been growing in service provider networks and the implementations should be capable of handling increase in LSP scale.

RFC 2961 specifies mechanisms to eliminate the reliance on periodic refresh and refresh timeout of RSVP messages, and enables a router to increase the message refresh interval to values much larger than the default 30 seconds defined in RFC 2205. However, the protocol extensions defined in RFC 4090 for supporting fast reroute (FRR) using bypass tunnels implicitly rely on short refresh timeouts to cleanup stale states.

In order to eliminate the reliance on refresh timeouts, the routers should unambiguously determine when a particular LSP state should be deleted. Coupling LSP state with the corresponding RSVP-TE signaling adjacencies as recommended in RSVP-TE Scaling Recommendations (draft-ietf-teas-rsvp-te-scaling-rec) will apply in scenarios other than RFC 4090 FRR using bypass tunnels. In scenarios involving RFC 4090 FRR using bypass tunnels, additional explicit tear down messages are necessary. Refresh-interval Independent RSVP FRR (RI-RSVP-FRR) extensions specified in this document consists of procedures to enable LSP state cleanup that are essential in scenarios not covered by procedures defined in RSVP-TE Scaling Recommendations.

## Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

## Table of Contents

1. Introduction.....	4
1.1. Motivation.....	4
2. Terminology.....	5
3. Problem Description.....	5
4. Solution Aspects.....	8
4.1. Signaling Handshake between PLR and MP.....	8
4.1.1. PLR Behavior.....	8
4.1.2. Remote Signaling Adjacency.....	10
4.1.3. MP Behavior.....	10
4.1.4. "Remote" state on MP.....	10
4.2. Impact of Failures on LSP State.....	11
4.2.1. Non-MP Behavior.....	12
4.2.2. LP-MP Behavior.....	12
4.2.3. NP-MP Behavior.....	12
4.2.4. Behavior of a Router that is both LP-MP and NP-MP...	13
4.3. Conditional Path Tear.....	14
4.3.1. Sending Conditional Path Tear.....	14
4.3.2. Processing Conditional Path Tear.....	14
4.3.3. CONDITIONS object.....	15
4.4. Remote State Teardown.....	16
4.4.1. PLR Behavior on Local Repair Failure.....	16
4.4.2. PLR Behavior on Resv RRO Change.....	17
4.4.3. LSP Preemption during Local Repair.....	17
4.4.3.1. Preemption on LP-MP after Phop Link failure....	17
4.4.3.2. Preemption on NP-MP after Phop Link failure....	18
4.5. Backward Compatibility Procedures.....	18
4.5.1. Detecting Support for Refresh interval Independent FRR	19
.....	
4.5.2. Procedures for backward compatibility.....	19
4.5.2.1. Lack of support on Downstream Node.....	19
4.5.2.2. Lack of support on Upstream Node.....	20
4.5.2.3. Incremental Deployment.....	20
5. Security Considerations.....	21
6. IANA Considerations.....	22
6.1. New Object - CONDITIONS.....	22
7. Normative References.....	22
8. Informative References.....	23
9. Acknowledgments.....	23
10. Contributors.....	23
11. Authors' Addresses.....	24

## 1. Introduction

RSVP-TE Fast Reroute [RFC4090] defines two local repair techniques to reroute label switched path (LSP) traffic over pre-established backup tunnel. Facility backup method allows one or more LSPs traversing a connected link or node to be protected using a bypass tunnel. The many-to-one nature of local repair technique is attractive from scalability point of view. This document enumerates facility backup procedures in RFC 4090 that rely on refresh timeout and hence make facility backup method refresh-interval dependent. The RSVP-TE extensions defined in this document will enhance the facility backup protection mechanism by making the corresponding procedures refresh-interval independent.

### 1.1. Motivation

Standard RSVP [RFC2205] maintains state via the generation of RSVP Path/Resv refresh messages. Refresh messages are used to both synchronize state between RSVP neighbors and to recover from lost RSVP messages. The use of Refresh messages to cover many possible failures has resulted in a number of operational problems.

- One problem relates to RSVP control plane scaling due to periodic refreshes of Path and Resv messages, another relates to the reliability and latency of RSVP signaling.
- An additional problem is the time to clean up the stale state after a tear message is lost. For more on these problems see Section 1 of RSVP Refresh Overhead Reduction Extensions [RFC2961].

The problems listed above adversely affect RSVP control plane scalability and RSVP-TE [RFC3209] inherited these problems from standard RSVP. Procedures specified in [RFC2961] address the above mentioned problems by eliminating dependency on refreshes for state synchronization and for recovering from lost RSVP messages, and by eliminating dependency on refresh timeout for stale state cleanup. Implementing these procedures allows to improve RSVP-TE control plane scalability. For more details on eliminating dependency on refresh timeout for stale state cleanup, refer to "Refresh Interval Independent RSVP" section in [TE-SCALE-REC].

However, the procedures specified in [RFC2961] do not fully address stale state cleanup for facility backup protection [RFC4090], as facility backup protection still depends on refresh timeouts for stale state cleanup. Thus [RFC2961] is insufficient to address the

problem of stale state cleanup when facility backup protection is used.

The procedures specified in this document, in combination with [RFC2961], eliminate facility backup protection dependency on refresh timeouts for stale state cleanup. These procedures, in combination with [RFC2961], fully address the above mentioned problem of RSVP-TE stale state cleanup, including the cleanup for facility backup protection.

The procedures specified in this document assume reliable delivery of RSVP messages, as specified in [RFC2961]. Therefore this document makes support for [RFC2961] a pre-requisite.

## 2. Terminology

The reader is assumed to be familiar with the terminology in [RFC2205], [RFC3209], [RFC4090] and [RFC4558].

Phop node: Previous-hop router along the label switched path

PPhop node: Previous-Previous-hop router along the LSP

LP-MP node: Merge Point router at the tail of Link-protecting bypass tunnel

NP-MP node: Merger Point router at the tail of Node-protecting bypass tunnel

TED: Traffic Engineering Database

Conditional PathTear: PathTear message containing a suggestion to a receiving downstream router to retain Path state if the receiving router is NP-MP

Remote PathTear: PathTear message sent from Point of Local Repair (PLR) to MP to delete state on MP if PLR had not reliably sent backup Path state before

## 3. Problem Description

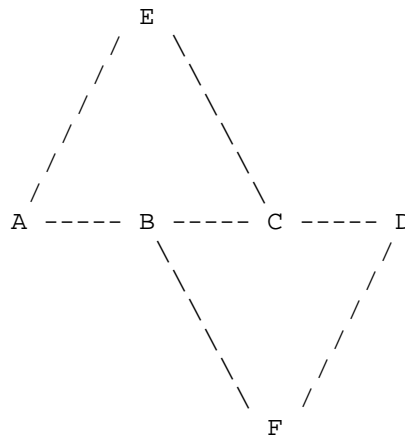


Figure 1: Example Topology

In the topology in Figure 1, consider a large number of LSPs from A to D transiting B and C. Assume that refresh interval has been configured to be large of the order of minutes and refresh reduction extensions are enabled on all routers.

Also assume that node protection has been configured for the LSPs and the LSPs are protected by each router in the following way

- A has made node protection available using bypass LSP A -> E -> C; A is the Point of Local Repair (PLR) and C is Node Protecting Merge Point (NP-MP)
- B has made node protection available using bypass LSP B -> F -> D; B is the PLR and D is the NP-MP
- C has made link protection available using bypass LSP C -> B -> F -> D; C is the PLR and D is the Link Protecting Merge Point (LP-MP)

In the above condition, assume that B-C link fails. The following is the sequence of events that is expected to occur for all protected LSPs under normal conditions.

1. B performs local repair and re-directs LSP traffic over the bypass LSP B -> F -> D.
2. B also creates backup state for the LSP and triggers sending of backup LSP state to D over the bypass LSP B -> F -> D.

3. D receives backup LSP states and merges the backups with the protected LSPs.
4. As the link on C, over which the LSP states are refreshed has failed, C will no longer receive state refreshes. Consequently the protected LSP states on C will time out and C will send tear down message for all LSPs. As each router should consider itself as a Merge Point, C will time out the state only after waiting for an additional duration equal to refresh timeout.

While the above sequence of events has been described in [RFC4090], there are a few problems for which no mechanism has been specified explicitly.

- If the protected LSP on C times out before D receives signaling for the backup LSP, then D would receive PathTear from C prior to receiving signaling for the backup LSP, thus resulting in deleting the LSP state. This would be possible at scale even with default refresh time.
- If upon the link failure C is to keep state until its timeout, then with long refresh interval this may result in a large amount of stale state on C. Alternatively, if upon the link failure C is to delete the state and send PathTear to D, this would result in deleting the state on D, thus deleting the LSP. D needs a reliable mechanism to determine whether it is MP or not to overcome this problem.
- If head-end A attempts to tear down LSP after step 1 but before step 2 of the above sequence, then B may receive the tear down message before step 2 and delete the LSP state from its state database. If B deletes its state without informing D, with long refresh interval this could cause (large) buildup of stale state on D.
- If B fails to perform local repair in step 1, then B will delete the LSP state from its state database without informing D. As B deletes its state without informing D, with long refresh interval this could cause (large) buildup of stale state on D.

The purpose of this document is to provide solutions to the above problems which will then make it practical to scale up to a large number of protected LSPs in the network.

#### 4. Solution Aspects

The solution consists of five parts.

- Utilize MP determination mechanism specified in [SUMMARY-FRR] that enables the PLR to signal availability of local protection to MP. In addition, introduce PLR and MP procedures to establish Node-ID hello session between the PLR and the MP to detect router failures and to determine capability. See section 4.1 for more details. This part of the solution re-uses some of the extensions defined in [SUMMARY-FRR] and [TE-SCALE-REC], and the subsequent sub-sections will list the extensions in these drafts that are utilized in this document.
- Handle upstream link or node failures by cleaning up LSP states if the node has not found itself as MP through the MP determination mechanism. See section 4.2 for more details.

The combination of "path state" maintained as Path State Block (PSB) and "reservation state" maintained as Reservation State Block (RSB) forms an individual LSP state on an RSVP-TE speaker.

- Introduce extensions to enable a router to send tear down message to downstream router that enables the receiving router to conditionally delete its local state. See section 4.3 for more details.
- Enhance facility protection by allowing a PLR to directly send tear down message to MP without requiring the PLR to either have a working bypass LSP or have already signaled backup LSP state. See section 4.4 for more details.
- Introduce extensions to enable the above procedures to be backward compatible with routers along the LSP path running implementation that do not support these procedures. See section 4.5 for more details.

##### 4.1. Signaling Handshake between PLR and MP

###### 4.1.1. PLR Behavior

As per the procedures specified in RFC 4090, when a protected LSP comes up and if the "local protection desired" flag is set in the SESSION\_ATTRIBUTE object, each node along the LSP path attempts to make local protection available for the LSP.



- If the "node protection desired" flag is set, then the node tries to become a PLR by attempting to create a NP-bypass LSP to the NNhop node avoiding the Nhop node on protected LSP path. In case node protection could not be made available after some time out, the node attempts to create a LP-bypass LSP to Nhop node avoiding only the link that protected LSP takes to reach Nhop
- If the "node protection desired" flag is not set, then the PLR attempts to create a LP-bypass LSP to Nhop node avoiding the link that the protected LSP takes to reach Nhop

With regard to the PLR procedures described above and that are specified in RFC 4090, this document specifies the following additional procedures.

- While selecting the destination address of the bypass LSP, the PLR SHOULD attempt to select the router ID of the NNhop or Nhop node. If the PLR and the MP are in same area, then the PLR may utilize the TED to determine the router ID from the interface address in RRO (if NodeID is not included in RRO). If the PLR and the MP are in different IGP areas, then the PLR SHOULD use the NodeID address of NNhop MP if included in the RRO of RESV. If the NP-MP in a different area has not included NodeID in RRO, then the PLR SHOULD use NP-MP's interface address present in the RRO. The PLR SHOULD use its router ID as the source address of the bypass LSP. The PLR SHOULD also include its router ID as the NodeID in PATH RRO unless configured explicitly not to include NodeID.
- In parallel to the attempt made to create NP-bypass or LP-bypass, the PLR SHOULD initiate a Node-ID based Hello session to the NNhop or Nhop node respectively to establish the RSVP-TE signaling adjacency. This Hello session is used to detect MP node failure as well as determine the capability of the MP node. If the MP sets I-bit in CAPABILITY object [TE-SCALE-REC] carried in Hello message corresponding to NodeID based Hello session, then the PLR SHOULD conclude that the MP supports refresh-interval independent FRR procedures defined in this document.
- If the bypass LSP comes up, then the PLR SHOULD include Bypass Summary FRR Association object and triggers PATH to be sent. If Bypass Summary FRR Association object is included in PATH message, then the encoding rules specified in [SUMMARY-FRR] MUST be followed.

#### 4.1.2. Remote Signaling Adjacency

A NodeID based RSVP-TE Hello session is one in which NodeID is used in source and destination address fields in RSVP Hello. [RFC4558] formalizes NodeID based Hello messages between two routers. This document extends NodeID based RSVP Hello session to track the state of RSVP-TE neighbor that is not directly connected by at least one interface. In order to apply NodeID based RSVP-TE Hello session between any two routers that are not immediate neighbors, the router that supports the extensions defined in the document SHOULD set TTL to 255 in the NodeID based Hello messages exchanged between PLR and MP. The default hello interval for this NodeID hello session SHOULD be set to the default specified in [TE-SCALE-REC].

In the rest of the document the term "signaling adjacency", or "remote signaling adjacency" refers specifically to the RSVP-TE signaling adjacency.

#### 4.1.3. MP Behavior

When the NNhop or Nhop node receives the triggered PATH with a "matching" Bypass Summary FRR Association object, the node should consider itself as the MP for the PLR IP address "corresponding" to the Bypass Summary FRR Association object. The matching and ordering rules of Bypass Summary FRR Association specified in [SUMMARY-FRR] SHOULD be followed by implementations supporting this document.

In addition to the above procedures, the node SHOULD check the presence of remote signaling adjacency with PLR (this check is needed to detect network being partitioned). If a matching Bypass Summary FRR Association object is found in PATH and the RSVP-TE signaling adjacency is present, the node concludes that the PLR will undertake refresh-interval independent FRR procedures specified in this document. If the PLR has included NodeID in PATH RRO, then that NodeID is the remote neighbor address. Otherwise, the PLR's interface address in RRO will be the remote neighbor address. If a matching Bypass Summary FRR Association object is included by PPhop node, then it is NP-MP. If a matching Bypass Summary FRR Association object is included by Phop node, it concludes it is LP-MP.

#### 4.1.4. "Remote" state on MP

Once a router concludes it is MP for a PLR running refresh-interval independent FRR procedures, it SHOULD create a remote path state for

the LSP. The "remote" state is identical to the protected LSP path state except for the difference in RSVP\_HOP object. The RSVP\_HOP object in "remote" Path state contains the address that the PLR uses to send NodeID hello messages to MP.

The MP SHOULD consider the "remote" path state automatically deleted if:

- MP later receives a PATH with no matching Bypass Summary FRR Association object corresponding to the PLR RRO, or
- Node signaling adjacency with PLR goes down, or
- MP receives backup LSP signaling from PLR or
- MP receives PathTear, or
- MP deletes the LSP state on local policy or exception event

Unlike the normal path state that is either locally generated on Ingress or created from PATH message from Phop node, the "remote" path state is not signaled explicitly from PLR. The purpose of "remote" path state is to enable the PLR to explicitly tear down path and reservation states corresponding to the LSP by sending tear message for the "remote" path state. Such message tearing down "remote" path state is called "Remote PathTear."

The scenarios in which "Remote" PathTear is applied are described in Section 4.4 - Remote State Teardown.

#### 4.2. Impact of Failures on LSP State

This section describes the procedures for routers on the LSP path for different kinds of failures. The procedures described on detecting RSVP control plane adjacency failures do not impact the RSVP-TE graceful restart mechanisms ([RFC3473], [RFC5063]). If the router executing these procedures act as helper for neighboring router, then the control plane adjacency will be declared as having failed after taking into account the grace period extended for neighbor by the helper.

Immediate node failures are detected from the state of NodeID hello sessions established with immediate neighbors. [TE-SCALE-REC] recommends each router to establish NodeID hello sessions with all its immediate neighbors. PLR or MP node failure is detected from the

state of remote signaling adjacency established according to Section 4.1.2 of this document.

#### 4.2.1. Non-MP Behavior

When a router detects Phop link or Phop node failure and the router is not an MP for the LSP, then it SHOULD send Conditional PathTear (refer to Section "Conditional PathTear" below) and delete PSB and RSB states corresponding to the LSP.

#### 4.2.2. LP-MP Behavior

When the Phop link for an LSP fails on a router that is LP-MP for the LSP, the LP-MP SHOULD retain PSB and RSB states corresponding to the LSP till the occurrence of any of the following events.

- Node-ID signaling adjacency with Phop PLR goes down, or
- MP receives normal or "Remote" PathTear for PSB, or
- MP receives ResvTear RSB.

When a router that is LP-MP for an LSP detects Phop node failure from Node-ID signaling adjacency state, the LP-MP SHOULD send normal PathTear and delete PSB and RSB states corresponding to the LSP.

#### 4.2.3. NP-MP Behavior

When a router that is NP-MP for an LSP detects Phop link failure, or Phop node failure from Node-ID signaling adjacency, the router SHOULD retain PSB and RSB states corresponding to the LSP till the occurrence of any of the following events.

- Remote Node-ID signaling adjacency with PPhop PLR goes down, or
- MP receives normal or "Remote" PathTear for PSB, or
- MP receives ResvTear for RSB.

When a router that is NP-MP does not detect Phop link or node failure, but receives Conditional PathTear from the Phop node, then the router SHOULD retain PSB and RSB states corresponding to the LSP till the occurrence of any of the following events.

- Remote Node-ID signaling adjacency with PPhop PLR goes down, or

- MP receives normal or "Remote" PathTear for PSB, or
- MP receives ResvTear for RSB.

Receiving Conditional PathTear from the Phop node will not impact the "remote" state from the PLR. Note that Phop node would send Conditional PathTear if it was not an MP.

In the example topology in Figure 1, assume C & D are NP-MP for PLRs A & B respectively. Now when A-B link fails, as B is not MP and its Phop link signaling adjacency has failed, B will delete LSP state (this behavior is required for unprotected LSPs - Section 4.2.1). In the data plane, that would require B to delete the label forwarding entry corresponding to the LSP. So if B's downstream nodes C and D continue to retain state, it would not be correct for D to continue to assume itself as NP-MP for PLR B.

The mechanism that enables D to stop considering itself as NP-MP and delete "remote" path state is given below.

1. When C receives Conditional PathTear from B, it decides to retain LSP state as it is NP-MP of PLR A. C also SHOULD check whether Phop B had previously signaled availability of node protection. As B had previously signaled NP availability in its PATH RRO, C SHOULD remove SUMMARY\_FRR\_BYPASS\_ASSOCIATION sub-object corresponding to B from the RRO and trigger PATH to D.
2. When D receives triggered PATH, it realizes that it is no longer NP-MP and so deletes the "remote" path state. D does not propagate PATH further down because the only change is in PATH RRO SUMMARY\_FRR\_BYPASS\_ASSOCIATION sub-object corresponding to B.

#### 4.2.4. Behavior of a Router that is both LP-MP and NP-MP

A router may be both LP-MP as well as NP-MP at the same time for Phop and PPhop nodes respectively of an LSP. If Phop link fails on such node, the node SHOULD retain PSB and RSB states corresponding to the LSP till the occurrence of any of the following events.

- Both Node-ID signaling adjacencies with Phop and PPhop nodes go down, or
- MP receives normal or "Remote" PathTear for PSB, or

- MP receives ResvTear for RSB.

If a router that is both LP-MP and NP-MP detects Phop node failure, then the node SHOULD retain PSB and RSB states corresponding to the LSP till the occurrence of any of the following events.

- Remote Node-ID signaling adjacency with PPhop PLR goes down, or
- MP receives normal or "Remote" PathTear for PSB, or
- MP receives ResvTear for RSB.

#### 4.3. Conditional Path Tear

In the example provided in the Section 4.2.5 "NP-MP Behavior on PLR link failure", B deletes PSB and RSB states corresponding to the LSP once B detects its link to Phop went down as B is not MP. If B were to send PathTear normally, then C would delete LSP state immediately. In order to avoid this, there should be some mechanism by which B can indicate to C that B does not require the receiving node to unconditionally delete the LSP state immediately. For this, B SHOULD add a new optional object called CONDITIONS object in PathTear. The new optional object is defined in Section 4.3.3. If node C also understands the new object, then C SHOULD delete LSP state only if it is not an NP-MP - in other words C SHOULD delete LSP state if there is no "remote" PLR state on C.

##### 4.3.1. Sending Conditional Path Tear

A router that is not an MP for an LSP SHOULD delete PSB and RSB states corresponding to the LSP if Phop link or Phop Node-ID signaling adjacency goes down (Section 4.2.1). The router SHOULD send Conditional PathTear if the following are also true.

- Ingress has requested node protection for the LSP, and
- PathTear is not received from upstream node

##### 4.3.2. Processing Conditional Path Tear

When a router that is not an NP-MP receives Conditional PathTear, the node SHOULD delete PSB and RSB states corresponding to the LSP, and process Conditional PathTear by considering it as normal PathTear. Specifically, the node SHOULD NOT propagate Conditional PathTear downstream but remove the optional object and send normal PathTear downstream.

When a node that is an NP-MP receives Conditional PathTear, it SHOULD NOT delete LSP state. The node SHOULD check whether the Phop node had previously included Bypass Summary FRR Association object in PATH. If the object had been included previously by Phop, then the node processing Conditional PathTear from Phop SHOULD remove the corresponding object and trigger PATH downstream.

If Conditional PathTear is received from a neighbor that has not advertised support (refer to Section 4.5) for the new procedures defined in this document, then the node SHOULD consider the message as normal PathTear. The node SHOULD propagate normal PathTear downstream and delete LSP state.

#### 4.3.3. CONDITIONS object

As any implementation that does not support Conditional PathTear SHOULD ignore the new object but process the message as normal PathTear without generating any error, the Class-Num of the new object SHOULD be 10bbbbbb where 'b' represents a bit (from Section 3.10 of [RFC2205]).

The new object is called as "CONDITIONS" object that will specify the conditions under which default processing rules of the RSVP-TE message SHOULD be invoked.

The object has the following format:

```

+-----+-----+-----+-----+-----+-----+-----+-----+
|          Length          |   Class   |   C-type   |
+-----+-----+-----+-----+-----+-----+-----+
|                               Reserved                               |M|
+-----+-----+-----+-----+-----+-----+-----+

```

Length

This contains the size of the object in bytes and should be set to eight.

Class

To be assigned

C-type

1

M bit

If M-bit is set to 1, then the PathTear message SHOULD be processed based on the condition if the receiver router is a Merge Point or not.

If M-bit is set to 0, then the PathTear message SHOULD be processed as normal PathTear message.

#### 4.4. Remote State Teardown

If the Ingress wants to tear down the LSP because of a management event while the LSP is being locally repaired at a transit PLR, it would not be desirable to wait till backup LSP signaling to perform state cleanup. To enable LSP state cleanup when the LSP is being locally repaired, the PLR SHOULD send "remote" PathTear message instructing the MP to delete PSB and RSB states corresponding to the LSP. The TTL in "remote" PathTear message SHOULD be set to 255.

Consider node C in example topology (Figure 1) has gone down and B locally repairs the LSP.

1. Ingress A receives a management event to tear down the LSP.
2. A sends normal PathTear to B.
3. To enable LSP state cleanup, B SHOULD send "remote" PathTear with destination IP address set to that of D used in Node-ID signaling adjacency with D, and RSVP\_HOP object containing local address used in Node-ID signaling adjacency.
4. B then deletes PSB and RSB states corresponding to the LSP.
5. On D there would be a remote signaling adjacency with B and so D SHOULD accept the remote PathTear and delete PSB and RSB states corresponding to the LSP.

##### 4.4.1. PLR Behavior on Local Repair Failure

If local repair fails on the PLR after a failure, then this should be considered as a case for cleaning up LSP state from PLR to the Egress. PLR would achieve this using "remote" PathTear to clean up state from MP. If MP has retained state, then it would propagate PathTear downstream thereby achieving state cleanup. Note that in



the case of link protection, the PathTear would be directed to LP-MP node IP address rather than the Nhop interface address.

#### 4.4.2. PLR Behavior on Resv RRO Change

When a router that has already made NP available detects a change in the RRO carried in RESV message, and if the RRO change indicates that the router's former NP-MP is no longer present in the LSP path, then the router SHOULD send "Remote" PathTear directly to its former NP-MP.

In the example topology in Figure 1, assume A has made node protection available and C has concluded it is NP-MP. When the B-C link fails then implementing the procedure specified in Section 4.2.4 of this document, C will retain state till: remote NodeID control plane adjacency with A goes down, or PathTear or ResvTear is received for PSB or RSB respectively. If B also has made node protection available, B will eventually complete backup LSP signaling with its NP-MP D and trigger RESV to A with RRO changed. The new RRO of the LSP carried in RESV will not contain C. When A processes the RESV with a new RRO not containing C - its former NP-MP, A SHOULD send "Remote" PathTear to C. When C receives a "Remote" PathTear for its PSB state, C will send normal PathTear downstream to D and delete both PSB and RSB states corresponding to the LSP. As D has already received backup LSP signaling from B, D will retain control plane and forwarding states corresponding to the LSP.

#### 4.4.3. LSP Preemption during Local Repair

If an LSP is preempted when there is no failure along the path of the LSP, the node on which preemption occurs would send PathErr and ResvTear upstream and only delete the forwarding state and RSB state corresponding to the LSP. But if the LSP is being locally repaired upstream of the node on which the LSP is preempted, then the node SHOULD delete both PSB and RSB states corresponding to the LSP and send normal PathTear downstream.

##### 4.4.3.1. Preemption on LP-MP after Phop Link failure

If an LSP is preempted on LP-MP after its Phop or incoming link has already failed but the backup LSP has not been signaled yet, then the node SHOULD send normal PathTear and delete both PSB and RSB states corresponding to the LSP. As the LP-MP has retained LSP state because the PLR would signal the LSP through backup LSP signaling, preemption would bring down the LSP and the node would not be LP-MP any more requiring the node to clean up LSP state.

#### 4.4.3.2. Preemption on NP-MP after Phop Link failure

If an LSP is preempted on NP-MP after its Phop link has already failed but the backup LSP has not been signaled yet, then the node SHOULD send normal PathTear and delete PSB and RSB states corresponding to the LSP. As the NP-MP has retained LSP state because the PLR would signal the LSP through backup LSP signaling, preemption would bring down the LSP and the node would not be NP-MP any more requiring the node to clean up LSP state.

Consider B-C link goes down on the same example topology (Figure 1). As C is NP-MP for PLR A, C will retain LSP state.

1. The LSP is preempted on C.
  2. C will delete RSB state corresponding to the LSP. But C cannot send PathErr or ResvTear to PLR A because backup LSP has not been signaled yet.
  3. As the only reason for C having retained state after Phop node failure was that it was NP-MP, C SHOULD send normal PathTear to D and delete PSB state also. D would also delete PSB and RSB states on receiving PathTear from C.
  4. B starts backup LSP signaling to D. But as D does not have the LSP state, it will reject backup LSP PATH and send PathErr to B.
  5. B will delete its reservation and send ResvTear to A.
- #### 4.5. Backward Compatibility Procedures

The "Refresh interval Independent FRR" or RI-RSVP-FRR referred below in this section refers to the changes that have been proposed in previous sections. Any implementation that does not support them has been termed as "non-RI-RSVP-FRR implementation". The extensions proposed in [SUMMARY-FRR] are applicable to implementations that do not support RI-RSVP-FRR. On the other hand, changes proposed relating to LSP state cleanup namely Conditional and remote PathTear require support from one-hop and two-hop neighboring nodes along the LSP path. So procedures that fall under LSP state cleanup category SHOULD be turned on only if all nodes involved in the node protection FRR i.e. PLR, MP and intermediate node in the case of NP, support the extensions. Note that for LSPs requesting only link protection, the PLR and the LP-MP should support the extensions.

#### 4.5.1. Detecting Support for Refresh interval Independent FRR

An implementation supporting the extensions specified in previous sections (called RI-RSVP-FRR here after) SHOULD set the flag "Refresh interval Independent RSVP" or RI-RSVP in CAPABILITY object in Hello messages. The RI-RSVP flag is specified in [TE-SCALE-REC].

- As nodes supporting the extensions SHOULD initiate Node Hellos with adjacent nodes, a node on the path of protected LSP can determine whether its PPhop or NPhop neighbor supports RI-RSVP-FRR enhancements from the Hello messages sent by the neighbor.
- If a node attempts to make node protection available, then the PLR SHOULD initiate remote Node-ID signaling adjacency with NNhop. If the NNhop (a) does not reply to remote node Hello message or (b) does not set "Enhanced facility protection" flag in CAPABILITY object in the reply, then the PLR can conclude that NNhop does not support RI-RSVP-FRR extensions.
- If node protection is requested for an LSP and if (a) PPhop node has not included a matching Bypass Summary FRR Association object in PATH or (b) PPhop node has not initiated remote node Hello messages, then the node SHOULD conclude that PLR does not support RI-RSVP-FRR extensions. The details are described in the "Procedures for backward compatibility" section below.

Any node that sets the I-bit is set in its CAPABILITY object MUST also set Refresh-Reduction-Capable bit in common header of all RSVP-TE messages.

#### 4.5.2. Procedures for backward compatibility

The procedures defined hereafter are performed on a subset of LSPs that traverse a node, rather than on all LSPs that traverse a node. This behavior is required to support backward compatibility for a subset of LSPs traversing nodes running non-RI-RSVP-FRR implementations.

##### 4.5.2.1. Lack of support on Downstream Node

- If the NPhop does not support the RI-RSVP-FRR extensions, then the node SHOULD reduce the "refresh period" in TIME\_VALUES object carried in PATH to default small refresh default value.
- If node protection is requested and the NNhop node does not support the enhancements, then the node SHOULD reduce the "refresh

period" in TIME\_VALUES object carried in PATH to a small refresh default value.

If the node reduces the refresh time from the above procedures, it SHOULD also not send remote PathTear or Conditional PathTear messages.

Consider the example topology in Figure 1. If C does not support the RI-RSVP-FRR extensions, then:

- A and B SHOULD reduce the refresh time to default value of 30 seconds and trigger PATH
- If B is not an MP and if Phop link of B fails, B cannot send Conditional PathTear to C but SHOULD time out PSB state from A normally. This would be accomplished if A would also reduce the refresh time to default value. So if C does not support the RI-RSVP-FRR extensions, then Phop B and PPhop A SHOULD reduce refresh time to a small default value.

#### 4.5.2.2. Lack of support on Upstream Node

- If Phop node does not support the RI-RSVP-FRR extensions, then the node SHOULD reduce the "refresh period" in TIME\_VALUES object carried in RESV to default small refresh time value.
- If node protection is requested and the Phop node does not support the RI-RSVP-FRR extensions, then the node SHOULD reduce the "refresh period" in TIME\_VALUES object carried in PATH to default value.
- If node protection is requested and PPhop node does not support the RI-RSVP-FRR extensions, then the node SHOULD reduce the "refresh period" in TIME\_VALUES object carried in RESV to default value.
- If the node reduces the refresh time from the above procedures, it SHOULD also not execute MP procedures specified in Section 4.2 of this document.

#### 4.5.2.3. Incremental Deployment

The backward compatibility procedures described in the previous subsections imply that a router supporting the RI-RSVP-FRR extensions specified in this document can apply the procedures specified in the document either in the downstream or upstream direction of an LSP,

depending on the capability of the routers downstream or upstream in the LSP path.

- RI-RSVP-FRR extensions and procedures are enabled for downstream Path, PathTear and ResvErr messages corresponding to an LSP if link protection is requested for the LSP and the Nhop node supports the extensions
- RI-RSVP-FRR extensions and procedures are enabled for downstream Path, PathTear and ResvErr messages corresponding to an LSP if node protection is requested for the LSP and both Nhop & NNhop nodes support the extensions
- RI-RSVP-FRR extensions and procedures are enabled for upstream PathErr, Resv and ResvTear messages corresponding to an LSP if link protection is requested for the LSP and the Phop node supports the extensions
- RI-RSVP-FRR extensions and procedures are enabled for upstream PathErr, Resv and ResvTear messages corresponding to an LSP if node protection is requested for the LSP and both Phop and PPhop nodes support the extensions

For example, if an implementation supporting the RI-RSVP-FRR extensions specified in this document is deployed on all routers in particular region of the network and if all the LSPs in the network request node protection, then the FRR extensions will only be applied for the LSP segments that traverse the particular region. This will aid incremental deployment of these extensions and also allow reaping the benefits of the extensions in portions of the network where it is supported.

## 5. Security Considerations

This security considerations pertaining to [RFC2205], [RFC3209] and [RFC5920] remain relevant.

This document extends the applicability of Node-ID based Hello session between immediate neighbors. The Node-ID based Hello session between PLR and NP-MP may require the two routers to exchange Hello messages with non-immediate neighbor. So, the implementations SHOULD provide the option to configure Node-ID neighbor specific or global authentication key to authentication messages received from Node-ID neighbors. The network administrator MAY utilize this option to enable RSVP-TE routers to authenticate Node-ID Hello messages received with TTL greater than 1. Implementations SHOULD also

provide the option to specify a limit on the number of Node-ID based Hello sessions that can be established on a router supporting the extensions defined in this document.

## 6. IANA Considerations

### 6.1. New Object - CONDITIONS

RSVP Change Guidelines [RFC3936] defines the Class-Number name space for RSVP objects. The name space is managed by IANA.

IANA registry: RSVP Parameters

Subsection: Class Names, Class Numbers, and Class Types

A new RSVP object using a Class-Number from 128-183 range called the "CONDITIONS" object is defined in Section 4.3 of this document. The Class-Number from 128-183 range will be allocated by IANA.

## 7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3209] Awduche, D., "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.
- [RFC4090] Pan, P., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, May 2005.
- [RFC2961] Berger, L., "RSVP Refresh Overhead Reduction Extensions", RFC 2961, April 2001.
- [RFC2205] Braden, R., "Resource Reservation Protocol (RSVP)", RFC 2205, September 1997.
- [RFC4558] Ali, Z., "Node-ID Based Resource Reservation (RSVP) Hello: A Clarification Statement", RFC 4558, June 2006.
- [RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching Signaling Resource Reservation Protocol-Traffic Engineering Extensions", RFC 3473, January 2003.
- [RFC5063] Satyanarayana, A., "Extensions to GMPLS Resource Reservation Protocol Graceful Restart", RFC5063, October 2007.

[RFC3936] Kompella, K. and J. Lang, "Procedures for Modifying the Resource reSerVation Protocol (RSVP)", BCP 96, RFC 3936, October 2004.

[TE-SCALE-REC] Vishnu Pavan Beeram et. al, "Implementation Recommendations to improve scalability of RSVP-TE Deployments", draft-ietf-teas-rsvp-te-scaling-rec (work in progress)

[SUMMARY-FRR] Mike Tallion et. al, "RSVP-TE Summary Fast Reroute Extensions for LSP Tunnels", draft-mtaillon-mpls-summary-frr-rsvpte (work in progress)

## 8. Informative References

[RFC5439] Yasukawa, S., "An Analysis of Scaling Issues in MPLS-TE Core Networks", RFC 5439, February 2009.

[RFC5920] Fang, L., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.

## 9. Acknowledgments

We are very grateful to Yakov Rekhter for his contributions to the development of the idea and thorough review of content of the draft. Thanks to Raveendra Torvi and Yimin Shen for their comments and inputs.

## 10. Contributors

Markus Jork  
Juniper Networks  
Email: mjork@juniper.net

Harish Sitaraman  
Juniper Networks  
Email: hsitaraman@juniper.net

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

Ebben Aries  
Juniper Networks

Email: [exa@juniper.net](mailto:exa@juniper.net)

Mike Tallion  
Cisco Systems Inc.  
Email: [mtallion@cisco.com](mailto:mtallion@cisco.com)

#### 11. Authors' Addresses

Chandra Ramachandran  
Juniper Networks  
Email: [csekar@juniper.net](mailto:csekar@juniper.net)

Ina Minei  
Google, Inc  
[inaminei@google.com](mailto:inaminei@google.com)

Dante Pacella  
Verizon  
Email: [dante.j.pacella@verizon.com](mailto:dante.j.pacella@verizon.com)

Tarek Saad  
Cisco Systems Inc.  
Email: [tsaad@cisco.com](mailto:tsaad@cisco.com)





MPLS Working Group  
INTERNET-DRAFT  
Intended Status: Proposed Standard  
Expires: September 14, 2017

Santosh Esale  
Raveendra Torvi  
Juniper Networks

Luyuan Fang  
Microsoft

Luay Jalil  
Verizon

March 13, 2017

Fast Reroute for Node Protection in LDP-based LSPs  
draft-esale-mpls-ldp-node-frr-05

Abstract

This document describes procedures to support node protection for unicast Label Switched Paths (LSPs) established by Label Distribution Protocol (LDP). In order to protect a node N, the Point of Local Repair (PLR) of N must discover the Merge Points (MPs) of node N such that traffic can be redirected to them in case of node N failure. Redirecting the traffic around the failed node N depends on existing point-to-point LSPs originated from the PLR to the MPs while bypassing the protected node N. The procedures described in this document are topology independent in a sense that they provide node protection in any topology so long as there is an alternate path in the network that avoids the protected node.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at

<http://www.ietf.org/lid-abstracts.html>

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

## Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
1.1 Abbreviations . . . . .	4
3. Merge Point (MP) Discovery . . . . .	4
4. Constructing Bypass LSPs . . . . .	5
5. Obtaining Label Mapping from MP . . . . .	6
6. Forwarding Considerations . . . . .	6
7. Synergy with node protection in mLDP . . . . .	7
8. Security Considerations . . . . .	7
9. IANA Considerations . . . . .	7
10. Acknowledgements . . . . .	7
11. Normative References . . . . .	7
12. Informative References . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

This document describes procedures to support node protection for unicast Label Switched Paths (LSPs) established by Label Distribution Protocol (LDP) [RFC5036]. In order to protect a node N, the Point of Local Repair (PLR) of N must discover the Merge Points (MPs) of node N such that traffic can be redirected to them in case of node N

failure. Redirecting the traffic around the failed node N depends on existing explicit path Point-to-Point (P2P) LSPs originated from the PLR LSR to the MPs while bypassing node N. The procedures to setup these P2P LSPs are outside the scope of this document, but one option is to use RSVP-TE based techniques [RFC3209] to accomplish it. Finally, sending traffic from the PLR to the MPs requires the PLR to obtain FEC-label bindings from the MPs. The procedures described in this document relies on Targeted LDP (tLDP) session [RFC5036] for the PLR to obtain such FEC-Label bindings.

The procedure described in this document assumes the use of platform-wide label space. The procedures for node protection described in this document fall into the category of local protection. The procedures described in this document apply to LDP LSPs bound to either an IPv4 or IPv6 Prefix FEC element. The procedures described in this document are topology independent in a sense that they provide node protection in any topology so long as there is a alternate path in the network that avoids the protected node. Thus these procedures provide topology independent fast reroute.

### 1.1 Abbreviations

PLR: Point of Local Repair - the LSR that redirects the traffic to one or more Merge Point LSRs.

MP: Merge Point. Any LSR on the LDP-signaled (multi-point to point) LSP, provided that the path from that LSR to the egress of that LSP is not affected by the failure of the protected node.

tLDP: A targeted LDP session is an LDP session between non-directly connected LSRs, established using the LDP extended discovery mechanism.

FEC: Forwarding equivalence class.

IGP: Interior Gateway Protocol.

BR: Border Router.

### 3. Merge Point (MP) Discovery

For a given LSP that traverses the PLR, the protected node N, and a particular neighbor of the protected node, we'll refer to this neighbor as the "next next-hop". Note that from the PLR's perspective the protected node N is the next hop for the FEC associated with that LSP. Likewise, from the protected node's perspective the next next-hop is the next hop for that FEC. If for a given <LSP, PLR, N> triplet the next next-hop is in the same routing subdomain (area) as the PLR, then that next next-hop acts as the MP for that triplet. For a given LSP traversing a PLR and the node protected by the PLR, the PLR discovers its next next-hops (MPs) that are in the same routing subdomain (IGP area) as the PLR from IGP shortest path first (SPF) calculations. The discovery of next next-hop, depending on an implementation, may not involve any additional SPF, above and beyond what will be needed by either ISIS or OSPF anyway, as the next next-hop, just like the next-hop, is a by-product of SPF computation.

Also, the PLR may discover all possible MPs from either its traffic engineering database or link state database. Some implementations MAY need appropriate configuration to populate the traffic engineering database. The traffic engineering database is populated by routing protocols such as ISIS and OSPF or configured statically.

If for a given <LSP, PLR, N> triplet the node protected by the PLR is an Border Router (BR), then the PLR and the next next-hop may end up in different routing subdomain. This could happen when an LSP

traversing the PLR and the protected node does not terminate in the same routing subdomain as the PLR. In this situation the PLR may not be able to determine the next next-hop from shortest path first (SPF) calculations, and thus may not be able to use the next next-hop as the MP. In this scenario the PLR uses an "alternative" BR as the MP, where an alternative BR is defined as follows. For a given LSP that traverses the PLR and the (protected) BR, an alternative BR is defined as any BR that advertises into PLR's own routing subdomain reachability to the FEC associated with the LSP.

Note that even if a PLR protects an BR, for some of the LSPs traversing the PLR and the BR, the next next-hops may be in the same routing subdomain as the PLR, in which case these next next-hops act as MPs for these LSPs. Note that even if the protected node is not an BR, if an LSP traversing the PLR and the protected node does not terminate in the same routing subdomain as the PLR, then for this LSP the PLR MAY use an alternative BR (as defined earlier), rather than the next next-hop as the MP. When there are several candidate BRs for alternative BR, the LSR MUST select one BR. The algorithm used for the alternative BR selection is a local matter but one option is to select the BR per FEC based on shortest path from PLR to the BR.

#### 4. Constructing Bypass LSPs

As mentioned before, redirecting traffic around the failed node N depends on existing explicit path Point-to-Point (P2P) LSPs originated from the PLR to the MPs while bypassing node N. Let's refer to these LSPs as "bypass LSPs". While the procedures to signal these bypass LSPs are outside the scope of this document, this document assumes use of RSVP-TE LSPs [RFC3209] to accomplish it. Once a PLR that protects a given node N discovers the set of MPs associated with itself and the protected node, at the minimum the PLR MUST (automatically) establish bypass LSPs to all these MPs. The bypass LSPs MUST be established prior to the failure of the protected node.

One could observe that if the protected node is not an BR and the PLR does not use alternative BR(s) as MP(s), then the set of all the IGP neighbors of the protected node forms a superset of the MPs. Thus it would be sufficient for the PLR to establish bypass LSPs with all the IGP neighbors of the protected node, even though some of these neighbors may not be MPs for any of the LSPs traversing the PLR and the protected node.

The bypass LSPs MUST avoid traversing the protected node, which means that the bypass LSPs are explicitly routed LSPs. Of course, using

RSVP-TE to establish bypass LSPs allows these LSPs to be explicitly routed. As a given router may act as an MP for more than one LSP traversing the PLR, the protected node, and the MP, the same bypass LSP will be used to protect all those LSPs.

## 5. Obtaining Label Mapping from MP

As mentioned before, sending traffic from the PLR to the MPs requires the PLR to obtain FEC-label bindings from the MPs. The solution described in this document relies on Targeted LDP (tLDP) session [RFC5036] for the PLR to obtain such mappings. Specifically, for a given PLR and the node protected by this PLR, at the minimum the PLR MUST (automatically) establish tLDP with all the MPs associated with this PLR and the protected node. These tLDP sessions MUST be established prior to the failure of the protected node. One could observe that if the protected node is not an BR and the PLR does not use alternative BR(s) as MP(s), then the set of all the IGP neighbors of the protected node forms a superset of the MPs. Thus it will be sufficient for the PLR to (automatically) establish tLDP session with all the IGP neighbors of the protected node - except the PLR - that are in the same area as the PLR, even though some of these neighbors may not be MPs for any of the LSPs traversing the PLR and the protected node.

At the minimum for a given tLDP peer the PLR MUST obtain FEC-label mapping for the FEC(s) for which the peer acts as an MP. The PLR MUST obtain this mapping before the failure of the protected node. To obtain this mapping for only these FECs and no other FECs that the peer may maintain, the PLR SHOULD rely on the LDP Downstream on Demand (DoD) procedures [RFC5036]. Otherwise, without relying on the DoD procedures, the PLR may end up receiving from a given tLDP peer FEC-label mappings for all the FECs maintained by the peer, even if the peer does not act as an MP for some of these FECs. If the LDP DoD procedures are not used, then for the purpose of the procedures specified in this draft the only label mappings that SHOULD be exchanged are for the Prefix FEC elements whose PreLen value is either 32 (IPv4), or 128 (IPv6); label mappings for the Prefix FEC elements with any other PreLen value SHOULD NOT be exchanged.

When a PLR has one or more BRs acting as MPs, the PLR MAY use the procedures specified in [draft-ietf-mpls-app-aware-tldp] to limit the set of FEC-label mappings received from non-BR MPs to only the mappings for the FECs associated with the LSPs that terminate in the PLR's own routing subdomain (area).

## 6. Forwarding Considerations

When a PLR detects failure of the protected node then rather than

swapping an incoming label with a label that the PLR received from the protected node, the PLR swaps the incoming label with the label that the PLR receives from the MP, and then pushes the label associated with the bypass LSP to that MP.

To minimize micro-loop during the IGP global convergence PLR may continue to use the bypass LSP during network convergence by adding small delay before switching to a new path.

#### 7. Synergy with node protection in mLDP

Both the bypass LSPs and tLDP sessions described in this document could also be used for the purpose of mLDP node protection, as described in [draft-ietf-mpls-mldp-node-protection].

#### 8. Security Considerations

The same security considerations apply as those for the base LDP specification, as described in [RFC5036].

#### 9. IANA Considerations

This document introduces no new IANA Considerations.

#### 10. Acknowledgements

We are indebted to Yakov Rekhter for many discussions on this topic. We like to thank Hannes Gredler, Aman Kapoor, Minto Jeyananth, Eric Rosen, Vladimir Blazhkun and Loa Andersson for through review of this document.

#### 11. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3209] D. Awduche, et al., "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC3209, Decembet 2001.
- [RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.
- [draft-ietf-mpls-app-aware-tldp] Esale, S., et al., "Application-aware Targeted LDP", draft-esale-mpls-app-aware-tldp, work in progress.

#### 12. Informative References



[RFC7715], IJ. Wijnands, et al., "Multipoint LDP (mLDP) Node Protection", RFC7715, January 2016.

Authors' Addresses

Santosh Esale  
Juniper Networks  
EMail: sesale@juniper.net

Raveendra Torvi  
Juniper Networks  
EMail: rtorvi@juniper.net

Luyuan Fang  
Microsoft  
Email: lufang@microsoft.com

Luay Jalil  
Verizon  
Email: luay.jalil@verizon.com

INTERNET-DRAFT  
Intended Status: Informational  
Expires: July 26, 2016

Luyuan Fang  
Deepak Bansal  
Microsoft  
Fabio Chiussi

Chandra Ramachandran  
Juniper Networks  
Ebben Aries  
Facebook  
Shahram Davari  
Broadcom  
Barak Gafni  
Mellanox  
Daniel Voyer  
Bell Canada  
Nabil Bitar  
Verizon

January 23, 2016

MPLS-Based Hierarchical SDN for Hyper-Scale DC/Cloud  
draft-fang-mpls-hsdn-for-hsdc-05

Abstract

This document describes Hierarchical SDN (HSDN), an architectural solution to scale the Data Center (DC) and Data Center Interconnect (DCI) networks to support tens of millions of physical underlay endpoints, while efficiently handling both Equal Cost Multi Path (ECMP) load-balanced traffic and any-to-any end-to-end Traffic Engineered (TE) traffic. HSDN achieves massive scale using surprisingly small forwarding tables in the network nodes. HSDN introduces a new paradigm for both forwarding and control planes, in that all paths in the network are pre-established in the forwarding tables and the labels can identify entire paths rather than simply destinations. The HSDN forwarding architecture is based on four main concepts: 1. Dividing the DC and DCI in a hierarchically-partitioned structure; 2. Assigning groups of Underlay Border Nodes in charge of forwarding within each partition; 3. Constructing HSDN MPLS label stacks to identify endpoints and paths according to the HSDN structure; and 4. Forwarding using the HSDN MPLS labels.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

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

#### Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Introduction . . . . .	4
1.1. Terminology . . . . .	6
1.2. DC and DCI Reference Model . . . . .	8
2. Requirements . . . . .	10
2.1. MPLS-Based HSDN Design Requirements . . . . .	10
2.2. Hardware Requirements . . . . .	11
3. HSDN Architecture - Forwarding Plane . . . . .	11
3.1. Hierarchical Underlay Partitioning . . . . .	12
3.2. Underlay Partition Border Nodes . . . . .	14
3.2.1. UPBN and UPBG Naming Convention . . . . .	17
3.2.2. HSDN Label Stack . . . . .	17
3.2.3. HSDN Design Example . . . . .	18
3.3. MPLS-Based HSDN Forwarding . . . . .	20
3.3.1 Non-TE Traffic . . . . .	21

3.3.2 TE Traffic . . . . .	23
4. Scalability Analysis . . . . .	24
4.1. LFIB Sizing - ECMP . . . . .	24
4.2. LFIB Sizing - TE . . . . .	25
5. HSDN Label Stack Assignment Scheme . . . . .	26
6. HSDN Architecture - Control Plane . . . . .	28
6.1. The SDN Approach . . . . .	28
6.2. HSDN Distributed Control Plane . . . . .	29
7. Security Considerations . . . . .	29
8. IANA Considerations . . . . .	29
9. Acknowledgments . . . . .	30
10. Contributors . . . . .	30
11. References . . . . .	31
11.1 Normative References . . . . .	31
11.2 Informative References . . . . .	31
Authors' Addresses . . . . .	32

## 1. Introduction

With the growth in the demand for cloud services, the end-to-end cloud network, which includes Data Center (DC) and Data Center Interconnect (DCI) networks, has to scale to support millions to tens of millions of underlay network endpoints. These endpoints can be bare-metal servers, virtualized servers, or physical and virtualized network functions and appliances.

The scalability challenge is twofold: 1. Being able to scale using low-cost network nodes while achieving high resource utilization in the network; and 2. Being able to scale at low operational and computational complexity while supporting both Equal-Cost Multi-Path (ECMP) load-balanced traffic and any-to-any Traffic Engineering (TE) traffic.

Being able to scale at low cost requires to avoid the potential explosion of the routing tables in the network nodes as the number of underlay network endpoints increases. Current commodity switches have relatively small routing and forwarding tables. For example, the typical Forwarding Information Base (FIBs) and Label Forwarding Information Base (LFIBs) tables in current low-cost network nodes contain 16K or 32K entries. These small sizes are clearly insufficient to support entries for all the endpoints in the hyper-scale cloud. Address aggregation is used to ameliorate the problem, but the scalability challenges remain, since the dynamic and elastic environment in the DC/cloud often brings the need to handle finely granular prefixes in the network in order to support Virtual Machine (VM) and Virtualized Network Function (VNF) mobility.

Other factors contribute to the FIB/LFIB explosion. For example, in a typical DC using a fat Clos topology, even the support of ECMP load balancing may become an issue if the individual outgoing paths belonging to an ECMP group carry different outgoing labels, since a single destination may contribute multiple entries in the tables.

Another key scalability issue to resolve is the complexity of certain desired functions that should be supported in the network, the most prominent one being TE. Currently, any-to-any server-to-server TE in the DC/DCI is simply unfeasible, as path computation and bandwidth allocation at scale, an NP-complete problem, becomes rapidly unmanageable. Furthermore, the forwarding state needed in the network nodes for TE tunnels contributes in a major way to the explosion of the LFIBs, since each TE tunnel corresponds to an entry in the tables.

Other major scalability issues are related to the efficient creation, management, and use of tunnels, for example the configuration of

protection paths for fast restoration.

Many additional scalability issues in terms of operational and computational complexity need to be resolved in order to scale the control plane and the network state. In particular, the controller-centric approach of Software Defined Networks (SDNs), which is increasingly accepted as "the way to build the next generation clouds," still needs to be demonstrated to be scalable to the levels required in the hyper-scale DC and cloud.

Finally, the underlay network architecture should offer certain capabilities to facilitate the support of the demands of the overlay network.

In this document, we present Hierarchical SDN (HSDN), a set of solutions for all these scalability challenges in the underlay network, both in the forwarding and in the control plane.

Although HSDN can be used with any forwarding technology, including IPv4 and IPv6, it has been designed to leverage Multi Protocol Label Switching (MPLS)-based forwarding [RFC3031], using label stacks [RFC3032] constructed according to the HSDN structure. This document therefore describes MPLS-based HSDN. Here, we describe end-to-end (host-to-host) MPLS-based HSDN, where the entire HSDN label stacks from source to destination are imposed at the server's Network Interface Cards (NICs), and thus all the IP lookups are confined to the network edges. However, MPLS-based HSDN does not need to be end-to-end, since label imposition could happen instead at the network nodes (e.g., at the Top-of-Rack (ToR) switches), or intermediate lookups in the network could be introduced, or even a combination of MPLS and IP forwarding could be deployed as part of the HSDN network.

The HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology. In this document, we assume a MPLS-based overlay technology using a Virtual Network (VN) Label, which is encapsulated in the HSDN label stack. However the description can be easily generalized to any overlay technology, such as BGP/MPLS IP VPNs [RFC4364], EVPN [RFC7432], VXLAN [RFC7348], NVGRE [RFC7637], Geneve [I-D.draft-gross-geneve], and other technologies.

HSDN achieves massive scale using surprisingly small LFIBs in the network nodes, while supporting both ECMP load-balanced traffic and any-to-any end-to-end TE traffic [HSDNSOSR15]. HSDN also brings important simplifications in the control plane and in the architecture of the SDN controller.

The HSDN architecture and operation is characterized by two fundamental properties. First, all paths in the network are pre-

established in the forwarding tables. Second, the HSDN labels can identify entire paths or groups of paths rather than simply destinations.

These two properties radically simplify establishing and handling tunnels. In addition to optimally handling both ECMP and Non-Equal Cost Multi Path load balancing, HSDN enables any-to-any, end-to-end, server-to-server TE at scale. With HSDN, the "cost" of establishing a tunnel is essentially eliminated, since the "tunnels" are pre-established in the network, and the TE task becomes one of path assignment and bandwidth allocation to the flows. As a larger portion of the traffic can be engineered effectively, the network can be run at a higher utilization using comparatively smaller buffers at the nodes.

The HSDN forwarding architecture in the underlay network is based on four main concepts: 1. Dividing the DC and DCI in a hierarchically-partitioned structure; 2. Assigning groups of Underlay Border Nodes in charge of forwarding within each partition; 3. Constructing HSDN MPLS label stacks to identify the end points according to the HSDN structure; and 4. Forwarding using the HSDN MPLS labels.

HSDN is designed to allow the physical decoupling of control and forwarding, and have the LFIBs configured by a controller according to a full SDN approach. The controller-centric approach is described in this document. In this context, "MPLS forwarding" in HSDN simply means using MPLS labels to forward the packets, since there is no need for label distribution protocols.

However, the HSDN control plane can also be built using a hybrid approach, in which a routing or label distribution protocol is used to distribute the labels, in conjunction with a SDN controller. This hybrid approach may be particularly useful during technology migration. The use of BGP Labeled Unicast (BGP-LU) for label distribution and LFIB configuration in a HSDN architecture is described in [I-D.fang-idr-bgplu-for-hsdn].

### 1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Term	Definition
-----	-----
BGP	Border Gateway Protocol
BGP-LU	Border Gateway Protocol Labeled Unicast
DC	Data Center

DCGW	DC Gateway (Border Leaf)
DCI	Data Center Interconnect
DID	Destination Identifier
ECMP	Equal Cost MultiPathing
FIB	Forwarding Information Base
HSDN	Hierarchical SDN
LDP	Label Distribution Protocol
LFIB	Label Forwarding Information Base
LN	Leaf Node
MPLS	Multi-Protocol Label Switching
NIC	Network Interface Card
PID	Path Identifier
SDN	Software Defined Network
SN	Spine Node
SVR	Server
UP	Underlay Partition
UPBG	Underlay Partition Border Group
UPBN	Underlay Partition Border Node
TE	Traffic Engineering
ToR	Top-of-Rack switch
TR	Top-of-Rack switch (used in figures)
VN	Virtual Network
VM	Virtual Machine
VNF	Virtualized Network Function
WAN	Wide Area Network

In this document, we also use the following terms.

- o End device: A physical device attached to the DC/DCI network. Examples of end devices include bare metal servers, virtualized servers, network appliances, etc.
- o Level: A layer in the hierarchy of underlay partitions in the HSDN architecture.
- o Overlay Network (ON): A virtualized network that provides Layer 2 or Layer 3 virtual network services to multiple tenants. It is implemented over the underlay network.
- o Path Label (PL): A label used for MPLS-based HSDN forwarding in the underlay network.
- o Row: A row of racks where end devices reside in a DC.
- o Tier: One of the layers of network nodes in a Clos-based topology.
- o Underlay Network (UN): The physical network that provides the connectivity among physical end devices. It provides transport for



the overlay network traffic.

- o Underlay Partition (UP): A logical portion of the underlay network designed according to the HSDN architecture. Underlay partitions are arranged in a hierarchy consisting of multiple levels.
- o VN Label (VL): A label carrying overlay network traffic. It is encapsulated in the underlay network in a stack of path labels constructed according to the HSDN forwarding scheme.

## 1.2. DC and DCI Reference Model

Here we show the typical structure of the DC and DCI, which we use in the rest of this document to describe the HSDN architecture. We also introduce a few commonly used terms to assist in the explanation.

Figure 1 illustrates multiple DCs interconnected by the DCI/WAN.

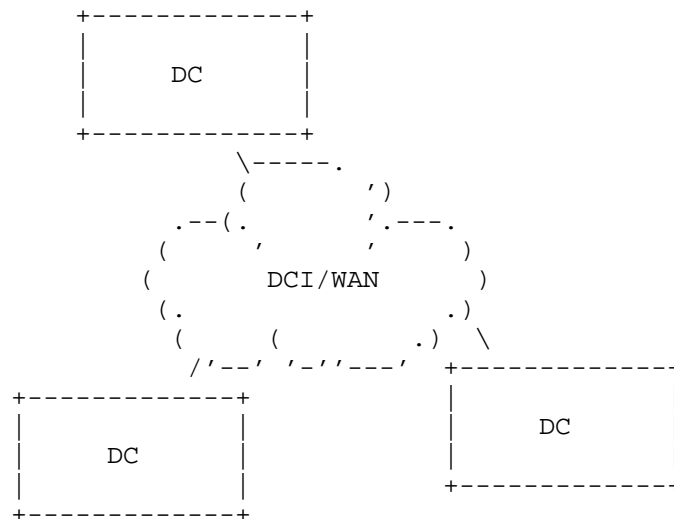


Figure 1. DCI/WAN interconnecting multiple DCs.

Figure 2 below illustrates the typical structure of a Clos-based DC fabric.

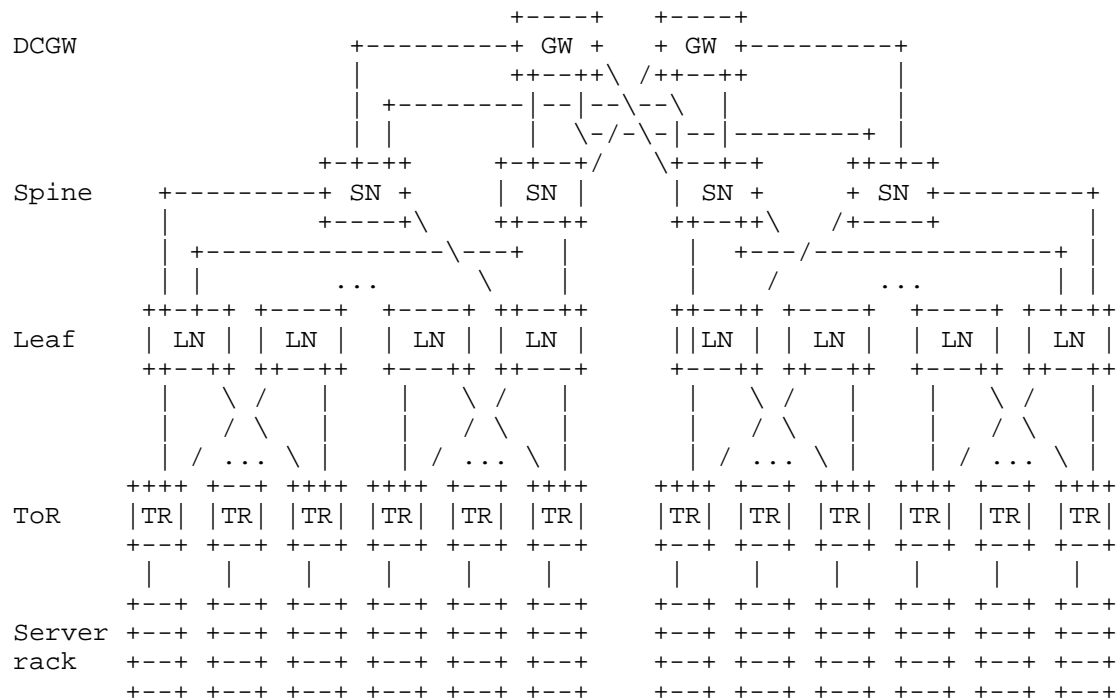


Figure 2. Typical Clos-based DC fabric topology.

Note: Not all nodes and links are shown in Figure 2.

The DC fabric shown in Figure 2 uses what is known as a spine and leaf architecture with a multi-stage Clos-based topology interconnecting multiple tiers of network nodes. The DC Gateways (DCGWs) connect the DC to the DCI/WAN. The DCGW connect to the Spine Nodes (SNs), which in turn connect to the Leaf Nodes (LFs). The Leaf Nodes connect to the ToRs. Each ToR typically resides in a rack (hence the name) accommodating a number of servers connected to their respective ToR. The servers may be bare metal or virtualized.

Each tier of switches and the connectivity between switches is designed to offer a desired capacity and provide sufficient bandwidth to the servers and end devices.

Figure 2 is not meant to represent the precise topology of the DC. In fact, the precise topology and connectivity between the tiers of switches depends on the specific design of the DC. More or less tiers of switches (spines or leafs) or asymmetric topologies, not shown in the figure, may be used. A precise description of the possible topologies and related design criteria is out of the scope of this

document.

What is relevant for this document is the fact that a typical large-scale DC topology does not have all the tiers fully connected to the adjacent tier (i.e., not all network nodes in a tier are necessarily connected to all network nodes in the adjacent tiers). This is especially true for the tiers closer to the endpoints, and is due to the sheer number of connections and devices (in other words, in a large, fat Clos there are too many network nodes in some tiers for all network nodes to connect to one another), and to the physical constraints of the DC (i.e., the network nodes may be located physically apart in separate rooms or buildings, and full connectivity may become too costly).

In a typical DC, the racks of servers are physically organized in "clusters" of racks, and dedicated banks of leaf switches may serve the ToRs in each cluster. For example, the racks may be physically placed in rows of racks, and a cluster of racks may correspond to a portion of a row, an entire row, or multiple rows of racks. Indeed, the leaf nodes are sometimes called "middle (or end) of the row switches" because they are physically located in a rack in the middle (or end) of a row of racks of servers. In turn, leaf nodes may also be organized in "zones" (we use "clusters" and "zones" as generic terms, but other terms may be used in the industry to refer to similar concepts), and banks of spines may be assigned to serve each zone. For example, a zone may include all the banks of leaf nodes that are in a room or in a building in the DC.

The actual connectivity is typically organized following an aggregation/multiplexing connectivity architecture that consolidates traffic from the edges into the leafs and spines, while allowing for over-subscription in order to strike a reasonable trade-off between cost and available capacity. The connectivity between each tier may use some form of shuffle-exchange topology that attempts to "mix" the available paths while taking in account the physical constraints.

The key observation is that it is impractical, uneconomical, and ultimately unnecessary to use a fully connected Clos-based topology in a large scale DC. Because of the physical constraints, the topology of a large DC is not a flat, fully-connected Clos, but rather has a certain hierarchy. The HSDN architecture recognizes this fact, and uses it to dramatically simplify forwarding and control planes using an approach that is also hierarchical.

## 2. Requirements

### 2.1. MPLS-Based HSDN Design Requirements

The following are the key design requirements for MPLS-based HSDN solutions.

- 1) MUST support millions to tens of millions of underlay network endpoints in the DC/DCI.
- 2) MUST use very small LFIB sizes (e.g., 16K or 32K LFIB entries) in all network nodes.
- 3) MUST support both ECMP load-balanced traffic and any-to-any, end-to-end, server-to-server TE traffic.
- 4) MUST support ECMP traffic load balancing using a single forwarding entry in the LFIBs per ECMP group.
- 5) MUST require IP lookup only at the network edges.
- 6) MUST support encapsulation of overlay network traffic, and support any network virtualization overlay technology.
- 7) MUST support control plane using both full SDN controller approach, and traditional distributed control plane approach using any label distribution protocols.

## 2.2. Hardware Requirements

The following are the hardware requirements to support HSDN.

- 1) The server NICs MUST be able to push a HSDN label stack consisting of as many path labels as levels in the HSDN hierarchical partition (e.g., 3 path labels).
- 2) The network nodes MUST support MPLS forwarding.
- 3) The network nodes MUST be able perform ECMP load balancing on packets carrying a label stack consisting of as many path labels as levels in the HSDN hierarchical partition, plus one or more VN label/header for the overlay network (e.g., 3 path labels + 1 VN label/header). For example, if the hash function used for ECMP forwarding is based on the IP 5-tuple, as is often the case, this requirement implies that the network nodes MUST be able to lookup the 5-tuple inside up to four labels.

## 3. HSDN Architecture - Forwarding Plane

As mentioned above, a primary design requirement for HSDN is to enable scalability of the forwarding plane to tens of millions of network endpoints using very small LFIB sizes in all network nodes in

the DC/DCI, while supporting both ECMP and any-to-any server-to-server TE traffic.

The driving principle of the HSDN forwarding plane is "divide and conquer" by partitioning the forwarding task into local and independent forwarding. When designed properly, such an approach enables extreme horizontal scaling of the DC/DCI.

HSDN is based on four concepts:

- 1) Dividing the underlay network in a hierarchy of partitions;
- 2) Assigning groups of Underlay Partition Border Nodes (UPBN) to each partition, in charge of forwarding within the corresponding partition;
- 3) Constructing HSDN label stacks for the endpoint Forward Equivalency Classes (FECs) in accordance with the underlay network partition hierarchy;
- 4) Configuring the LFIBs in all network nodes and forwarding using the label stacks.

As explained in Section 3.3.1, the HSDN label stacks can be used to identify entire paths to each endpoint, rather than simply the destination endpoint itself. As a matter of fact, the HSDN solution is meant to be configured with all possible paths in the network pre-established in the LFIBs in the network nodes. In this case, a FEC per path to each endpoint is defined. However, because of the way the HSDN architecture is designed, the required local number of entries in the LFIB of each network node remains surprisingly small.

In this section, we explain in detail each of these concepts. Scalability analysis for both ECMP load-balanced and TE traffic is presented in Section 4. In Section 5, we describe a possible label stack assignment scheme for HSDN.

### 3.1. Hierarchical Underlay Partitioning

HSDN is based on dividing the DC/DCI underlay network into logical partitions arranged in a multi-level hierarchy.

The HSDN hierarchical partitioning is illustrated in Figure 3.

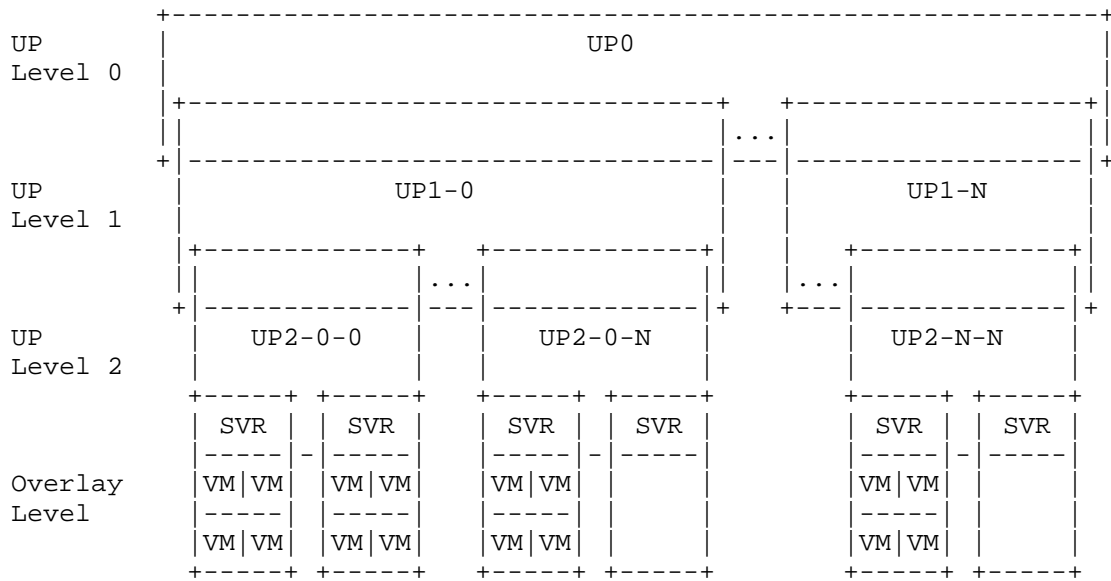


Figure 3. HSDN underlay network hierarchical partitioning of DC/DCI.

The hierarchy consists of multiple levels of Underlay Partitions (UPs). For simplicity, we describe HSDN using three levels of partitioning, but more or less levels can be used, depending on the size and architecture of the overall network, using similar design principles (as shown in Section 4, three levels of partitions are sufficient to achieve scalability to tens of millions servers using very small LFIBs).

The levels of partitions are nested into a hierarchical structure. At each level, the combination of all partitions covers the entire DC/DCI topology. In general, within each level, the UPs do not overlap, although there may be design scenarios in which overlapping UPs within a level may be used. The top level (Level 0) consists of a single underlay partition UP0 (the HSDN concept can be extended to multi-partitioned Level 0).

We use the following naming convention for the UPs:

- Partitions at Level *i* are referred to as UP<sub>*i*</sub> (e.g., UP0 for Level 0, UP1 for Level 1, UP2 for Level2, and so on).
- Within each level, partitions are identified by a rightmost sequential number (starting from 1) referring to the corresponding level and a set of sequential number(s) for each partition in a

higher level that the specific partition is nested into.

For example, at Level 1, there are N partitions, referred to as UP1-1 to UP1-N.

Similarly, at Level 2, there are M partitions for each Level 1 partitions, for a total of NxM partitions. For example, the Level 2 partitions nested into Level 1 partition UP1-1 are UP2-1-1 to UP2-1-M, while the ones nested into UP1-N are UP2-N-1 to UP2-N-M.

- Note that for simplicity in illustrating the partitioning, we assume a symmetrical arrangement of the partitions, where the number of partitions nested into each partition at a higher level is the same (e.g., all UP1 partitions have M UP2 partitions). In practice, this is rarely the case, and the naming convention can be adapted accordingly for different numbers of partitions nesting into each higher level partition (e.g., partition UP1-1 has M1 UP2 partitions, partition UP1-2 has M2 UP2 partitions, and so on).

The following considerations complete the description of Figure 3.

- o The servers (bare metal or virtualized) are attached to the bottom UP level (in our case, Level 2). A similar naming convention as the one used for the partitions may be used.
- o In Figure 3, we also show an additional Overlay Level. This corresponds to the virtualized overlay network (if any) providing Virtual Networks (VN) connecting Virtual Machines (VMs) and other overlay network endpoints. Overlay network traffic is encapsulated by the HSDN underlay network. As mentioned in the Introduction, the HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology, such as BGP/MPLS IP VPNs [RFC4364], EVPN [RFC7432], VXLAN [RFC7348], NVGRE [RFC7637], Geneve [I-D.draft-gross-geneve], and other technologies. A full description of the encapsulation of these technologies into the HSDN underlay label stack is out of scope of this document and will be addressed in a separate document.

The UPs are designed to contain one or more tiers of switches in the DC topology or nodes in the DCI. The key design criteria in defining the partitions at each level is that they need to follow the "natural" connectivity implemented in the DC/DCI topology. An example is given in Section 3.2.3 to further clarify how the partitions are designed.

### 3.2. Underlay Partition Border Nodes

Once the HSDN hierarchical partitioning is defined, Underlay

Partition Border Nodes (UPBNs) are assigned to each UP. This is illustrated in Figure 4.

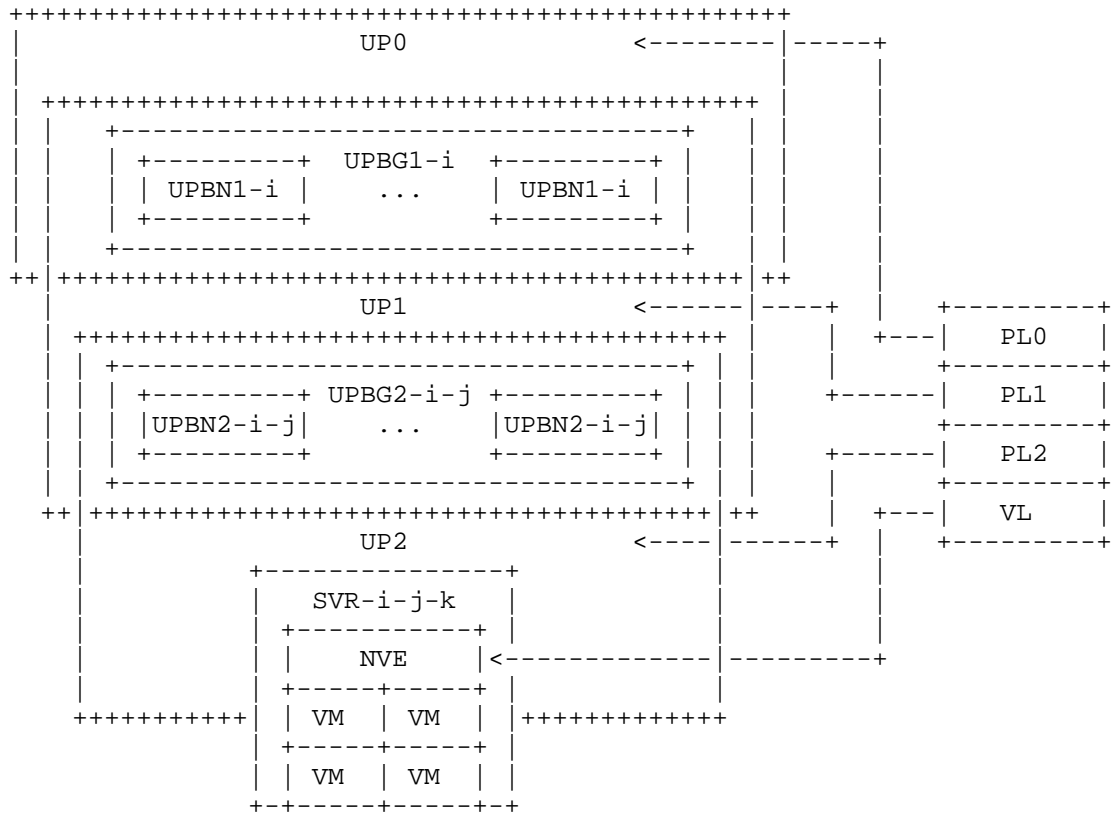


Figure 4. UBPNs, UBPGs, and label stack assignment.

The UPBNs serve as the connecting nodes between adjacent partitions. As such, the UPBNs belong to two partitions in adjacent levels in the hierarchy and they constitute the entry points for traffic from the higher level partition destined to the corresponding lower level partition (and vice-versa, they are the exit points for traffic from a lower level partition to a higher level partition). As such, they constitute the forwarding end destinations within each partition.

In order to provide sufficient capacity and support traffic load balancing between the levels in the hierarchy, multiple UPBNs are assigned to each partition. The UPBNs for each partition are grouped into an Underlay Partition Border Group (UPBG). As shown in Section 5, using an appropriate Label Stack Assignment scheme all UPBNs in a



UPBG can be made identical for ECMP traffic forwarding (i.e., the ECMP entries in the LFIBs in all UPBNs in a UPBG are identical). Thus, for ECMP traffic load balancing, all UPBNs belong to the same FEC as far as the higher level partition is concerned. For TE traffic, a desired UPBN within a UPBG group may need to be specified, and thus the UPBNs in a UPBG are not forwarding-wise equivalent.

In practice, the UPs are designed by finding the most advantageous way to partition the DC Clos-based topology and the DCI topology. As mentioned above, the connectivity of any large-scale DC is not fully flat, but rather contains some sort of hierarchical organization. Recognizing the hierarchy of the physical connectivity is an important starting point in the design of the partitions.

Within the DC, the UPBNs in each level are subsets of the network nodes in one of the tiers that form the multi-stage Clos architecture.

In general, in addition to the UPBNs, the UPs may internally contain tiers of network nodes that are not UPBNs. A specific design example to further illustrate the HSDN partitioning is provided in Section 3.2.3.

As explained in more detail in Section 3.3, for forwarding purposes, by partitioning the DC/DCI in this manner and using HSDN forwarding, the UPBNs need to have entries in their LFIBs only to reach destinations in the two partitions to which they belong to (i.e., their own corresponding lower-level partition and the higher-level partition to which they nested to). The network nodes inside the UPs only need to have entries in their LFIB to reach the destinations in their partition.

Similarly, in order to establish all possible paths in the entire network, the UPBNs need to have entries in their LFIBs only for all possible paths to the destinations in the two partitions to which they belong to.

From these considerations, a first design heuristic for choosing the partitioning structure is to keep the number of partitions nested at each level into the higher level relatively small for all levels. For the lowest level, the number of endpoints (servers) in each partition should also be kept to manageable levels.

Clearly, the design tradeoff is between the size and the number of partitions at each level. Although finding the optimal design choice may require a little trial-and-error computation of different options, fortunately, for most practical deployments, it is relatively simple to find a good tradeoff that achieves the desired scalability

to millions or tens of millions of endpoints.

### 3.2.1. UPBN and UPBG Naming Convention

We use a similar naming convention for the UPBNs and UPBGs as the one used for the UPs:

- UPBN<sub>i</sub> is a UPBN between partitions at Level(*i*) and Level(*i*-1). Similarly for UPBG.
- Within each level, the UPBNs are identified by a set of sequential number(s) equal to the corresponding sequential number(s) of the corresponding partition within that level.

For example, at Level 1, UPBN1-1 corresponds to partition UP1-1, and connects UP0 with UP1-1. UPBN1-N corresponds to partition UP1-N and connects UP0 with UP1-N, and so on. Similarly for UPBG.

At Level 2, UPBN2-1-1 corresponds to partition UP2-1-1 and connects UP1-1 with UP2-1-1, and so on. Similarly for UPBG.

Note that the UPBNs within an UPBGs can be further distinguished using an appropriate naming convention (for example, using an additional sequential number within the UPBG), which for simplicity is not shown here. This more granular naming convention is needed to configure the paths and the TE tunnels.

### 3.2.2. HSDN Label Stack

In MPLS-Based HSDN, an MPLS label stack is defined and used for forwarding. The key notion in HSDN is that the label stack is defined and the labels are assigned in accordance with the hierarchical partitioned structure defined above.

The label stack, shown in Figure 4 above, is constructed as follows.

- The label stack contains as many Path Labels (PLs) as levels in the partitioning hierarchy.
- Each PL in the label stack is associated to a corresponding level in the partition hierarchy and is used for forwarding at that level.

In the scenario of Figure 4, PL0 is associated to Level 0 and is used to forward to destinations in UP0, PL1 is associated to Level 1 and is used to forward to destinations in any UP1 partitions, and PL2 is associated to Level 2 and is used to forward to destinations in any UP2 partitions.

- A VN Label (VL) is also shown in the label stack in Figure 4. This label is associated to the Overlay Level and is used to forward in the overlay network. The VL is simply encapsulated in the label stack and transported in the HSDN underlay network. As mentioned above, the HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology, and thus the VL may be a label, a tag, or some other identifier, depending on the overlay technology used. The details of the VL encapsulation and processing for different overlay technologies are out of scope of this document.

Each endpoint in the DC/DCI is identified by a corresponding label stack. For a given endpoint, the label stack is constructed in such a way that the PLO specifies the UP1 to which the endpoint is attached to, the PL1 specifies the UP2 to which the endpoint is attached to, and the PL2 specifies the FEC in the UP2 corresponding to the endpoint.

The labels in the HSDN label stack can identify entire paths, rather than simply the end destination within the corresponding partition. This can be used to bring dramatic simplifications in handling tunnels and TE traffic in particular, as further explained in Section 3.3.2.

As mentioned above, in this draft we describe end-to-end MPLS-based HSDN forwarding, where the entire HSDN label stacks from the sources to the destinations are inserted at the server's NICs. In this scenario, the label stack imposed at a server points all the way to the end destination of the packet, which may be in a different DC. With any-to-any, end-to-end TE, the HSDN label stack identifies the entire path to the destination. For inter-DC traffic, there may be cases where the path through the remote DC would be preferably determined when the packet arrives at that DC, or when the packet leaves the source DC, so it may be desirable that part of the label stack be imposed inside the network rather than at the server. Nothing precludes this design choice, and a lookup may be added where desired in the HSDN network.

A scheme to assign the PL labels in the HSDN label stack is described in Section 5.

### 3.2.3. HSDN Design Example

We use an example to further explain the HSDN design criteria to define the hierarchically-partitioned structure of the DC/DCI. We use the same design example in the Scalability Analysis section (Section 4) to show the LFIB sizing with ECMP and TE traffic.

To summarize some of the design heuristics for the HSDN underlay partitions:

- The UPs should be designed to follow the "natural" connectivity topology in the DC/DCI.
- The number of partitions at each level nested into the higher level should be relatively small (since they are FEC entries in the LFIBs in the network nodes in the corresponding levels).
- The number of endpoints (servers) in each partition in the lowest level should be relatively small (since they are FEC entries in the LFIBs in the network nodes in the lowest level).
- The number of levels should be kept small (since it corresponds to the number of path labels in the stack).
- The number of tiers in each partition in each level should be kept small. This is due to the multiplicative fanout effect for TE traffic (explained in Section 4.2), which has a major impact on the LFIB size needed to support any-to-any server-to-server TE.

The HSDN forwarding plane design consists in finding the best tradeoff among these conflicting objectives. Although the optimal design choices ultimately depend on the specific deployment, fortunately, it is generally rather straightforward to identify design choices that can support scalability to millions or tens of millions of servers.

Here we describe a design example to illustrate that a three-level HSDN hierarchy is sufficient to scale the DC/DCI to tens of millions of servers.

With three levels, a possible design choice for the UP1s is to have each UP1 correspond to a DC. With this choice, the UP0 corresponds to the DCI and the UPBN1s are the DCGWs in each DC (the UPBG1s group the DCGWs in each DC).

Once the UP1s are chosen this way, a possible design choice for the UP2s is to have each UP2 correspond to a group of racks, where each group of racks may correspond to a portion of a row of racks, an entire row of racks, or multiple rows of racks. The specific best choice of how many racks should be in a group of racks corresponding to each UP2 ultimately depends on the specific connectivity in the DC and the number of servers per racks.

While precise numbers depend on the specific technologies used in each deployment, here and in the Scalability Analysis section

(Section 4) we want to give some ideas of the scaling capabilities of HSDN. For this purpose, we use some hypothetical yet reasonable numbers to characterize the partitioning design example.

Assume the following: a) 20 DCs connected via the DCI/WAN; b) 50 servers per rack; c) 20 racks per group of racks; d) 50 groups of racks per DC.

With these numbers, there are 500K servers per DC, for a total of 10M underlay network endpoints in the DC/DCI.

In the HSDN structure in this example, there are 20 UP1s, 500 UP2s per UP1, and 1000 servers per UP2.

### 3.3. MPLS-Based HSDN Forwarding

The hierarchically partitioned structure and the corresponding label stack are used in HSDN to scale the forwarding plane horizontally while using LFIBs of surprising small sizes in the network nodes.

As explained above, each label in the HSDN label stack is associated with one of the levels in the hierarchy and is used to forward to destinations in the underlay partitions at that level.

With HSDN, by superimposing a hierarchically-partitioned structure and using a label stack constructed according to such a structure, we are able to impose a forwarding scheme that is aggregated by construction. This translates in dramatic reductions in the size of the LFIBs in the network nodes, since each node only needs to know a limited portion of the forwarding space.

HSDN supports any label assignment scheme to generate the labels in the label stack. However, if a label assignment scheme that is consistent with the HSDN structure is used, additional simplifications of the LFIBs and the control plane can be achieved.

In Section 5 below, we present one example of such a scheme, where the labels in the label stack represent the "physical" location of the endpoint, expressed according to the HSDN structure. For TE traffic, the labels represent a specific path towards the desired destination through the HSDN structure.

In the Scalability Analysis section (Section 4) and in the Control Plane section (Section 6) we assume that such a Label Assignment scheme is used.

In the rest of this section, we describe the life of a packet in the HSDN DC/DCI. We use the specific design example described in Section

3.2.3 above to help in the explanation, but of course the forwarding would be similar for other design choices.

### 3.3.1 Non-TE Traffic

We first describe the behavior for ECMP load-balanced, non-TE traffic. In the HSDN DC/DCI, for a packet that needs to be forwarded to a specific endpoint in the underlay network, the outer label PL0 specifies which UP1 contains the endpoint. Let's refer to this UP1 as UP1-a. For ECMP traffic, the PL0 binding is with a FEC corresponding to the UPBG1-a associated with UP1-a. Note that all the endpoints reachable via UP1-a are forwarded using the same FEC entry for Level 0 in the hierarchical partitioning.

Once the packet reaches one of the network nodes UPBN1-a in the UPBG1-a group (the upstream network nodes perform ECMP load balancing, thus the packet may enter UP1-a via any of the UPBN1-a nodes), the PL0 is popped and the PL1 is used for forwarding in the UP1-a (to be precise, because of penultimate hop popping, it is the network node immediately upstream of the chosen UPBN1-a that pops the label P0).

The PL1 is used within UP1-a to reach the UP2 which contains the endpoint. Let's refer to this UP2 as UP2-a. In the UP2 network nodes the PL1 binding is with a FEC corresponding to the UPBG2-a associated with UP2-a. Similarly as above, note that all the endpoints reachable via UP2-a are forwarded using the same FEC entry for Level 1 in the hierarchical partitioning.

Once the packet reaches one of the network nodes UPBN2-a in the UPBG2-a group (once again, the upstream network nodes perform ECMP load balancing, so the packet may transit to any of the UPBN2-a nodes), the PL1 is popped and the PL2 is used for the rest of the forwarding (again, to be precise, the penultimate network node upstream of UPBN2-a is the one popping the PL1 label).

The PL2 is used within UP2-a to reach the desired endpoint. Note that the UPBN2 nodes and the network nodes in the UP2s have entries in their LFIBs only to reach endpoints within their UP2. They can reach endpoints in other UP2s by using a FEC entry corresponding to the UP2 containing the destination endpoint, identified by PL1.

The following two observations help in further clarifying the forwarding operation above.

- The PL0 is used for forwarding from the source to the UPBN1-a. For a packet originating from an endpoint attached to a certain UP2, say UP2-b, nested to a different UP1, say UP1-b, PL0 is used for

forwarding in all network nodes that the packet transits until it reaches the UPBN1-a. This includes network nodes in UP2-b and UP1-b (i.e., "on the way up" from UP2). It also includes one of the UPBN1-b nodes.

It is important to note, however, that the PL0 is not popped at the UPBN1-b, since it is used for forwarding to the destination UPBN1-a.

- It should be pointed out that an important requirement for HSDN is to achieve route optimization for ECMP traffic, meaning that the hierarchy should forward a packet from any source to any destination using the same number of hops and without introducing any additional latency compared to a flat architecture. For example, a packet originating from an endpoint in UP2,N,M, and destined to an endpoint in the same UP2,N,M should not be forwarded all the way to the highest level in the hierarchy and back, but should be forwarded to the desired endpoint by "turning it around" towards the destination at the first node in the UP2,N,M that contains an entry to that desired endpoint. Indeed, if the packet turns around at the proper node, it will go through the same number of hops as it would have gone through in a flat architecture. This should hold true even in the case where the UP1s and/or UP2s contain intermediate tiers of switches and the packet needs to be turned around in the intermediate nodes.

Route optimization is easily achieved in HSDN by simply having the packets only carry the portion of the label stack that is needed to reach the destination using the appropriate turn around node. Continuing with our example, the packet above only needs PL2 to be optimally forwarded, since it should never "go out" of UP2,N,M. Thus, PL0 and PL1 should not be included in its label stack, to avoid an unnecessary round trip up and down the DC through all the levels in the hierarchy. Similarly, a packet originating and terminating in the same UP1, but in different UP2s, only needs PL1 and PL2 to be forwarded.

In this case, a network node would have to process different labels for traffic going up and out the partition versus traffic staying in the partition ("going up" and "coming down" refer to the direction of traffic in Figure 3). Since the label spaces for the two path labels may overlap, ambiguity would result. Depending on the LFIB configuration, the two Most Significant Bits (MSBs) in each label in Figure 4 may be reserved for identifying the layer (i.e., whether the label is PL0, PL1, or PL2) and resolve ambiguity.

A better solution to achieve the same without using precious bits

in the labels is to use a "turn around entry" in the LFIBs, which flags that the packet needs to turn around at that node and the relevant label is not the outer label (as it would be for traffic going up or coming down, for which the outer label just needs to pass through), but is the one underneath (thus, the outer label needs to be popped to expose the relevant label). In our example, the packet destined to an endpoint in the same UP2,N,M of the originating server may carry a PL1 corresponding to the "turn around" label value and a PL2 corresponding to the desired endpoint within UP2, and does not need a PL0.

In the case of ECMP load-balanced non-TE traffic, the labels in the HSDN label stack identify ECMP groups for each destination in the corresponding partition. In this way, at each node in the partition, the outgoing label is the same for all paths belonging to the same ECMP group. A label allocation scheme for this is described in Section 5.

### 3.3.2 TE Traffic

Handling TE traffic in the hyper-scale DC/DCI presents major scalability challenges, since each TE tunnel contributes one entry in the forwarding tables, and the TE path and bandwidth allocation computation is a NP-complete problem.

HSDN introduces radical simplifications in establishing and handling tunnels, and in supporting TE in particular.

In HSDN, all paths in the network can be pre-established in the LFIBs. Because of the way the HSDN architecture is constructed, the number of entries that have to be stored in the local LFIB in each network node remains surprisingly small.

In this case, the labels in the HSDN stack identify entire paths, or groups of paths, to each destination in each partition, rather than just the destination itself.

With HSDN, since the "cost" of establishing a tunnel is essentially eliminated (all "tunnels" are pre-established in the network), and the TE task becomes one of path assignment and bandwidth allocation to the flows. Furthermore, the hierarchical structure of HSDN makes it possible to devise algorithms and heuristics for path and bandwidth allocation computation that operate largely independently in each partition, and are therefore computationally feasible even at large scale. A description of such algorithms is out of scope of this document. As a larger portion of the traffic can be engineered effectively, the network can be run at a higher utilization using comparatively smaller buffers at the nodes.



Since all paths can be accommodated in the LFIBs, HSDN makes it possible to support "TE Max Case" with small LFIB sizes. In TE Max Case, all sources are connected to all destinations (e.g., server to server) with TE tunnels, the tunnels using all possible distinct paths in the network. TE Max Case gives therefore an upper bound to the number of TE tunnels (and consequently, LFIB entries) in the network.

The fact that the LFIBs remain relatively small even when all possible paths are configured is the consequence of two desirable properties of HSDN.

First, since in HSDN the individual UPs are designed in such a way to be relatively small, the number of paths in each partition can be kept to a manageable number.

Second, the hierarchical structure of HSDN makes it possible to use the partitioning astutely to break the "TE Fanout Multiplicative Effect," which defines the number of paths to a destination, and can easily contribute to the LFIB explosion as the number of hops and the fanout of each hop to each destination in the network increases. As explained in Section 4.2, with the hierarchical structure, the TE Fanout Multiplicative Effect is only multiplicative within each level in the hierarchy. Thus, by properly designing the partitioning, the multiplicative effect can be kept to a manageable level.

In the case of TE traffic, the processing of the different labels in the label stack is similar to what described above for ECMP load-balanced non-TE traffic. However, the labels are bound to FECs identifying a specific path within each UPs that is traversed.

#### 4. Scalability Analysis

In this section, we compute the maximum size of the LFIBs for non-TE/ECMP traffic and any-to-any server-to-server TE traffic.

##### 4.1. LFIB Sizing - ECMP

For ECMP traffic, at each level, all destinations belonging to the same partition at a lower level are forwarded using the same FEC entry in the LFIB, which identifies the destination UPBG for that level, or the destination endpoint at the lower level. Since the UPs are designed in such a way to keep the number of destinations small in all UPs, and the network nodes only need to know how to reach destinations in their own UP and in the adjacent UP at the higher level in the hierarchy, this translates to the fact that hyper scale of the DC/DCI can be achieved with very small LFIB sizes in all the individual network nodes.

A detailed explanation of how the LFIB size can be computed in all the nodes of an HSDN network is given in [HSDNSOSR15]. The worst case for the LFIB size occurs at one of the network nodes that serve as UPBNs for one of the levels of UPs in the hierarchy. The level where the LFIB size occurs depend on the specific choice of the partitioning design.

#### 4.2. LFIB Sizing - TE

As noted above, TE traffic may add a considerable number of entries to LFIB, since it creates one new FEC per TE tunnel to each destination.

HSDN provides a solution to this problem, and in fact, HSDN can support any-to-any server-to-server "TE Max Case" with small LFIB sizes.

In a Clos Topology (the analysis can be extended to generic topologies), the number of paths in a UP with N destination can be easily computed. The number of paths (and the maximum number of LFIB entries) is equal to the products of the switch fanout in each tier traversed from the source to the destination in that UP. This is the "TE Fanout Multiplicative Effect" mentioned above, which is illustrated in Figure 5. Accordingly:

Total # LFIB Entries for TE Max Case =  $N * F_1 * F_2 * \dots * F_{(M-1)}$

Where  $F_i$  is the fanout of a switch in each tier traversed to the destination, M is the number of tiers in the UP, and N is the number of destinations in the UP.

Once again, by properly designing the UPs, the TE Fanout Multiplicative Effect can be kept under control, since the path computation is local for each of the UPs. HSDN breaks the multiplicative effect, since the TE Fanout Multiplicative Effect is multiplicative only within each UP, rather than in the entire network, and the "multiplication" restarts at each level of the hierarchy. A detailed description of the LFIB computation in all network nodes to support TE Max Case is given in [HSDNSOSR15].

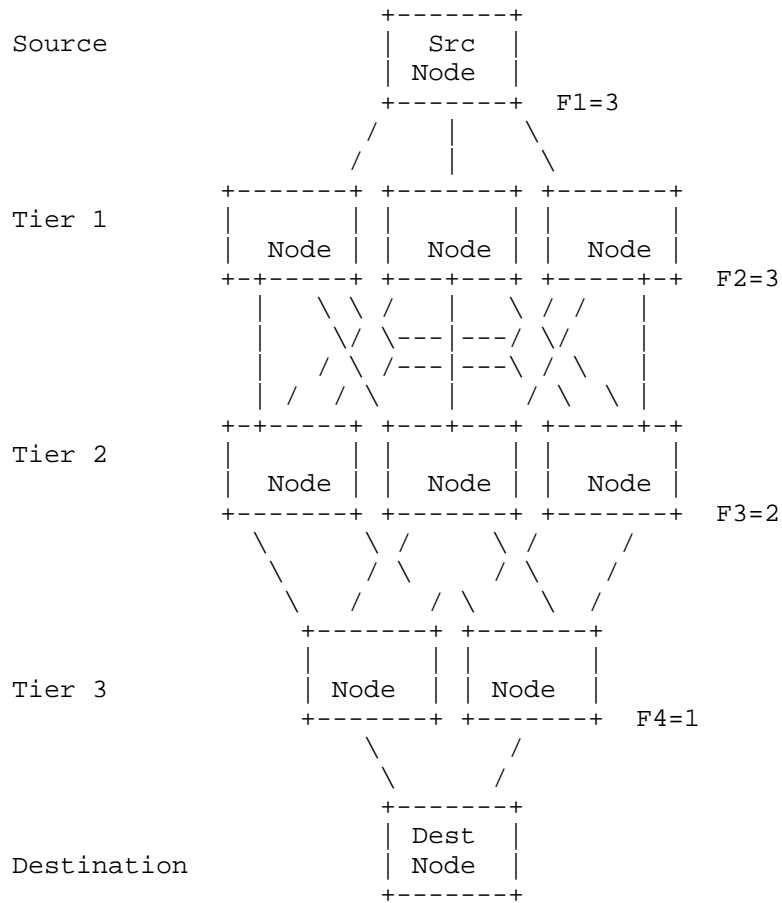


Figure 5. Fan out multiplicative effect with TE.

## 5. HSDN Label Stack Assignment Scheme

HSDN can use any scheme to assign the labels in the label stack. However, if a label assignment scheme which assigns labels in a way consistent with the HSDN structure, important simplifications can be achieved in the control plane and in the LFIBs.

For non-TE FECs, the HSDN label assignment scheme assigns labels according to the "physical" location of the endpoint in the HSDN structure. Continuing our design example from above, for an endpoint X in UP2-a, PL0 would identify the DC in which the endpoint is located, PL1 would identify the group of racks in which the endpoint is located within the DC, and PL2 would identify the endpoint within

the group of servers within the DC.

For TE FECs, the HSDN label assignment scheme assigns labels to identify a specific path in each UP that is traversed. In our example, for a specific TE tunnel to endpoint X, PL0 would identify the specific path that should be followed in the DCI, PL1 would identify the path that should be followed within the DC to reach the group of racks, and PL2 would identify the path to reach the endpoint within the group of racks (if there are multiple paths).

In order to assign labels to both non-TE traffic and TE traffic, HSDN uses a label format in which the labels are divided into two logical sub-fields, one identifying the destination within the UP, called Destination Identifier (DID), and one identifying the path, called Path Identifier (PID). The Path Identifier is only relevant for TE traffic, and can be zero for non-TE traffic. The HSDN Label format is illustrated in Figure 6.

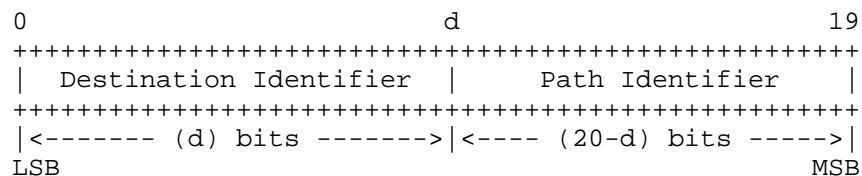


Figure 6. HSDN Label format.

In this label assignment scheme, the path labels associated with a partition are globally unique within that partition, meaning that different partitions at the same level can use the same label space. For PL0, the path labels are globally unique within the entire network, since there is only one UP0. Neither of these is a scaling limitation, since all partitions are relatively small.

The bits in the DID for each level must be sufficient to represent the distinct destinations that need to be known in the UPs at that level, and the bits in the PID for each level must be sufficient to represent all the distinct paths that need to be defined in the UPs at that level (closed-form expressions for both these numbers are given in [HSDNSOSR15]). In practice, this is not a significant scalability constraint: with three MPLS labels in the stack, partitioning architectures and label formats according to this scheme can be found to scale to tens of millions of servers.

Depending on the LFIB configuration, the two MSBs may be reserved for

identifying the level (i.e., whether the label is PL0, PL1, or PL2) to resolve ambiguity (not shown in Figure 6). Note, however, that this is not strictly necessary and the same function of identifying the level can be achieved by simply allocating "turn around" entries in the nodes, as explained in Section 3.3.1, so an individual node always sees the same label in the stack.

By properly designing the UPs, this label assignment scheme can support the desired scalability and the support of end-to-end TE traffic.

Note that by using this type of label assignment scheme important benefits can be achieved, including:

- The LFIBs become rather "static," since the FECs are tied to "physical" locations and paths, which change infrequently. This simplifies the use of the SDN approach to configure the LFIBs via a controller.
- All paths in each ECMP group use the same outgoing labels. This guarantees that a single LFIB entry can be used for each ECMP group.

The label stack needs to be imposed at the entry points. For an endpoint, this implies that the server NIC must be able to push a three-label stack of path labels (in addition to possibly push one additional VL label for the overlay network).

## 6. HSDN Architecture - Control Plane

HSDN has been designed to support the controller-centric SDN approach in a scalable fashion. HSDN also supports the traditional distributed control plane approach.

HSDN introduces important simplifications in the control plane and in the network state as well.

### 6.1. The SDN Approach

In the controller-centric SDN approach, the SDN controller configures the LFIBs in all the network nodes. With HSDN, the hierarchical partitioned structure offers a natural framework for a distributed implementation of the SDN controller, since the control plane in each UP is largely independent from other UPs. The individual UP control planes operate in parallel, with loose synchronization among one another.

Therefore, the HSDN control plane is logically partitioned in a way

that is consistent with the forwarding plane partitioning. Each UP is assigned a corresponding UP controller, which configures the LFIBs in the network nodes in the corresponding UP. The individual UP controllers communicate with one another to exchange the labels and construct the label stacks. In HSDN, configuring the LFIBs in the network nodes is not a difficult task, since the labels are static and configuration updates are needed only when the physical topology changes or endpoints are added or permanently removed, and thus they are not too frequent.

Each UP controller at the lowest level of the hierarchy is also in charge of providing the label stacks to the server's NICs in the corresponding partition. For this purpose, a number of label servers, which may also be arranged in a hierarchy, are used to provide the mappings between IP addresses and label stacks.

Redundancy is superimposed to the structure of UP controllers, with each UP controller shadowing UP controllers in other UPs.

The HSDN UP controllers may also be in charge of TE computation. HSDN TE path computation algorithms that perform for the most part partition-local computation (so the computation is also horizontally scalable) but still approach global optimality using inter-UP-controller synchronization at a different time scale, can be devised.

## 6.2. HSDN Distributed Control Plane

The HSDN control plane can also be built using a hybrid approach, in which a routing or label distribution protocol is used to distribute the labels, in conjunction with a controller. An example using BGP-LU [RFC3107] is presented in [I-D.fang-idr-bgplu-for-hsdn].

## 7. Security Considerations

When the SDN approach is used, the protocols used to configure the LFIBs in the network nodes MUST be mutually authenticated.

For general MPLS/GMPLS security considerations, refer to [RFC5920].

Given the potentially very large scale and the dynamic nature in the cloud/DC environment, the choice of key management mechanisms need to be further studied.

To be completed.

## 8. IANA Considerations

TBD.

## 9. Acknowledgments

We would like to acknowledge Yakov Rekhter for many discussions related to HSDN.

## 10. Contributors

Vijay Gill  
Salesforce  
Email: [vgill@salesforce.com](mailto:vgill@salesforce.com)

Linda Dunbar  
Huawei Technologies  
5430 Legacy Drive, Suite #175  
Plano, TX 75024  
Email: [linda.dunbar@huawei.com](mailto:linda.dunbar@huawei.com)

Andrew Qu  
MediaTek  
2860 Junction Ave.  
San Jose, CA 95134  
Email: [andrew.qu@mediatek.com](mailto:andrew.qu@mediatek.com)

Jeff Tantsura  
Ericsson  
200 Holger Way  
San Jose, CA 95134  
Email: [jeff.tantsura@ericsson.com](mailto:jeff.tantsura@ericsson.com)

Wen Wang  
Century Link  
2355 Dulles Corner Blvd.  
Herndon, VA 20171  
Email: [wen.wang@centurylink.com](mailto:wen.wang@centurylink.com)

Himanshu Shah  
Ciena  
3939 North 1st Street  
San Jose, CA 95112  
Email: [hshah@ciena.com](mailto:hshah@ciena.com)

Ramki Krishnan  
Dell  
Email: [ramki\\_krishnan@dell.com](mailto:ramki_krishnan@dell.com)

Iftekhar Hussain  
Infinera Corporation  
140 Caspian Ct,

Sunnyvale, CA 94089  
Email: ihussain@infinera.com

## 11. References

### 11.1 Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, May 2001.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.
- [RFC7432] Sajassi et al., "BGP MPLS Based Ethernet VPN", RFC 7432, February 2015.

### 11.2 Informative References

- [RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.
- [RFC7348] M. Mahalingam et al., "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, August 2014.
- [RFC7637] P. Garg et al., "NVGRE: Network Virtualization using Generic Routing Encapsulation", RFC 7637, Sept. 2015.
- [I-D.fang-idr-bgplu-for-hsdn] L. Fang et al., "BGP-LU for HSDN Label Distribution", draft-fang-idr-bgplu-for-hsdn-02 (work in progress), July 2015.
- [I-D.draft-gross-geneve] J. Gross et al., "Geneve: Generic Network Virtualization Encapsulation", draft-gross-geneve-02 (work in progress), October 2014.
- [HSDNSOSR15] L. Fang et al., "Hierarchical SDN for the Hyper-Scale,



Hyper-Elastic Data Center and Cloud", ACM SIGCOMM  
Symposium on SDN Research 2015, Santa Clara, CA, June  
2015.

## Authors' Addresses

Luyuan Fang  
Microsoft  
15590 NE 31st St.  
Redmond, WA 98052  
Email: lufang@microsoft.com

Deepak Bansal  
Microsoft  
15590 NE 31st St.  
Redmond, WA 98052  
Email: dbansal@microsoft.com

Fabio Chiusi  
Seattle, WA 98116  
Email: fabiochiussi@gmail.com

Chandra Ramachandran  
Juniper Networks  
Electra, Exora Business Park Marathahalli - Sarjapur Outer Ring Road  
Bangalore, KA 560103, India  
Email: csekar@juniper.net

Ebben Aries  
Facebook  
1601 Willow Road  
Menlo Park, CA 94025  
Email: exa@fb.com

Shahram Davari  
Broadcom  
3151 Zanker Road  
San Jose, CA 95134  
Email: davari@broadcom.com

Barak Gafni  
Mellanox  
6 Habarzel St.  
Tel Aviv, Israel  
Email: gbarak@mellanox.com

Daniel Voyer

Bell Canada  
Email: daniel.voyer@bell.ca

Nabil Bitar  
Verizon  
40 Sylvan Road  
Waltham, MA 02145  
Email: nabil.bitar@verizon.com

INTERNET-DRAFT  
Intended Status: Standards Track  
Expires: May 7, 2016

Luyuan Fang  
Microsoft  
Fabio Chiussi

Barak Gafni  
Mellanox  
Nabil Bitar  
Alcatel Lucent  
Zhenbin Li  
Huawei  
Rob Raszuk  
Bloomberg  
Andrew Qu  
MediaTek

November 4, 2015

MPLS Label Forwarding with No Swapping  
draft-fang-mpls-label-forwarding-no-swap-02

Abstract

This document defines MPLS label forwarding operation with no label swapping as a new MPLS label operation extension to the existing basic forwarding operation of label push, pop, and swap.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

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

## Copyright and License Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	4
3. Label Forwarding Operation as Defined in RFC3031 . . . . .	4
4. Label Forwarding with No Swap . . . . .	5
5. Proposed Text to Update RFC3031 . . . . .	6
6. Security Considerations . . . . .	7
7. IANA Considerations . . . . .	7
8. References . . . . .	7
8.1 Normative References . . . . .	7
8.2 Informative References . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

MPLS forwarding operation as defined in [RFC3031] has three basic operations that must be performed on the labels at the network nodes: push, swap, and pop. This document describes an additional operation action: label forwarding with no swap. Currently, using the same label as both incoming and outgoing label is typically achieved by "swapping" the incoming label with an identical outgoing label. In order to improve processing efficiency and memory usage reduction, a simple label forwarding operation with no swap is desirable. This operation should be specified in [RFC3031]

When MPLS Architecture [RFC3031] was defined, the three types of label operation were sufficient, since labels were always only locally meaningful. Label swap operation is performed at a Label Switched Router (LSR) which is not an MPLS edge node, while label push and pop can be performed at an MPLS edge node for label imposition and deposition. Penultimate hop popping can also be performed at the penultimate hop for improved efficiency when appropriate. Since the labels are assigned independently in distributed fashion in a non-traffic engineered basic MPLS networks, it is not possible nor necessary to coordinate the label assignment. Therefore, the label swapping function is sufficient and effective for LSR.

With the increased interests and large scale development of Software-Defined Networking (SDN), several new use cases have emerged where domain-wide labels are used. For example, central controller assigned MPLS labels become one of the options for MPLS based forwarding. Using a single label is used to traverse multiple hops along the Label Switched Path (LSP) become desirable. Relevant examples of the use of domain-wide labels are described in [I-D.fang-mpls-hsdn-for-hsdc], [I-D.ietf-spring-segment-routing], and [I-D.li-mpls-global-label-usecases]

Domain-wide labels do not need to be swapped at the switches. Therefore, in order to allow efficient handling of domain-wide labels, there is a need to extend the label operation with one more action type - forwarding with no swap.

The performance and memory efficiency can be increased by performing simple forwarding function than swapping the labels with the identical identifier. This does not otherwise change the fundamentals of MPLS architecture and label encoding as defined by [RFC3031] [RFC3032].

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

- o Incoming Label Map (ILM): It maps each incoming label to a set of NHLFEs. It is used when forwarding packets that arrive as labeled packets.
- o Label forwarding: A simple forwarding paradigm allowing streamlined forwarding of data by using labels to identify classes of data packets which are treated indistinguishably when forwarding without label swapping.
- o Label swap: The basic existing forwarding operation consisting of looking up an incoming label to determine the outgoing label encapsulation, port, and other data handling information.
- o Label swapping: A forwarding paradigm allowing streamlined forwarding of data by using labels to identify classes of data packets which are treated indistinguishably when forwarding.
- o Label Switched Path (LSP): The path through one or more LSRs at one level of the hierarchy followed by a packets in a particular forwarding equivalence class (FEC).
- o Label Switching Router (LSR): An MPLS node which is capable of forwarding native L3 packets.
- o MPLS edge node: An MPLS node that connects an MPLS domain with a node which is outside of the domain, either because it does not run MPLS, and/or because it is in a different domain. Note that if an LSR has a neighboring host which is not running MPLS, that that LSR is an MPLS edge node.
- o NHLFE: Next Hop Label Forwarding Entry
- o Software-Defined Networking (SDN): an architecture that decouples the network control and forwarding functions to enable the network control to be directly programmable and the underlying infrastructure to be abstracted for applications and network services.

## 3. Label Forwarding Operation as Defined in RFC3031

Section 3.10 in [RFC3031] states the following:

"The "Next Hop Label Forwarding Entry" (NHLFE) is used when forwarding a labeled packet. It contains the following information:

1. the packet's next hop
2. the operation to perform on the packet's label stack; this is one of the following operations:
  - a) replace the label at the top of the label stack with a specified new label
  - b) pop the label stack
  - c) replace the label at the top of the label stack with a specified new label, and then push one or more specified new labels onto the label stack.

..."

Section 3.13 in [RFC3031] states the following:

"Label swapping is the use of the following procedures to forward a packet. In order to forward a labeled packet, a LSR examines the label at the top of the label stack. It uses the ILM to map this label to an NHLFE. Using the information in the NHLFE, it determines where to forward the packet, and performs an operation on the packet's label stack. It then encodes the new label stack into the packet, and forwards the result."

It is clear that, in order to forward a labeled packet, [RFC3031] mandates to perform one of three possible operations: swap, pop, or swap/push.

Therefore, in order to forward a labeled packet with a label that does not need to be swapped according to the standard, the label at the top of the label stack must be swapped with an identical label.

Simply forwarding the labeled packet without swapping the label does not conform with [RFC3031].

#### 4. Label Forwarding with No Swap

Label forwarding is the use of the following procedures to forward a packet.

Same as in [RFC3031], in order to forward a labeled packet, a LSR examines the label at the top of the label stack. It uses the ILM to

map this label to an NHLFE. Using the information in the NHLFE, it determines where to forward the packet, and performs an operation on the packet's label stack.

Unlike in label swapping, label forwarding does not remove the incoming label and encodes the new label stack into the packet as in label swapping, it forwards the packet with the same label stack as the incoming stack, to the outgoing interface. Other processing may be involved in selecting the outgoing interface, for example, load balancing through IP deader hashing or use of Entropy label [RFC6790].

#### 5. Proposed Text to Update RFC3031

In order to allow efficient processing of domain-wide labels, which do not need to be swapped, we propose to add a fourth possible operation "Forward No Swap" to [RFC3031].

Accordingly, [RFC3031] should be modified as follows.

1. the packet's next hop
  2. the operation to perform on the packet's label stack; this is one of the following operations:
    - a) replace the label at the top of the label stack with a specified new label
    - b) pop the label stack
    - c) replace the label at the top of the label stack with a specified new label, and then push one or more specified new labels onto the label stack
    - d) keep the label at the top of the label stack unchanged.
- ...



## 6. Security Considerations

The MPLS label forwarding operation specified herein does not raise any security issues that are not already present in either the MPLS architecture [RFC3031] or in MPLS label encoding [RFC3032].

In addition, general MPLS and GMPLS considerations and MPLS security defense techniques are documented in [RFC5920].

## 7. IANA Considerations

None.

## 8. References

### 8.1 Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.
- [RFC6790] Kompella, K., et.al, "The Use of Entropy Labels in MPLS Forwarding", November 2012.

### 8.2 Informative References

- [RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.
- [I-D.fang-mpls-hsdn-for-hsdc] L. Fang, et al., "MPLS-Based Hierarchical SDN for Hyper-Scale DC/Cloud", draft-fang-mpls-hsdn-for-hsdc-04 (work in progress), July 2015.
- [I-D.ietf-spring-segment-routing] Filsfils, C. et al., "Segment Routing Architecture", draft-ietf-spring-segment-routing-06 (work in progress), October 2015.
- [I-D.li-mpls-global-label-usecases] Z. Li, et al., "Usecases of MPLS Global Label, draft-li-mpls-global-label-usecases-03 (work in progress), October 2015.

Authors' Addresses

Luyuan Fang  
Microsoft  
5600 148th Ave NE  
Redmond, WA 98052  
Email: lufang@microsoft.com

Fabio Chiusi  
Seattle, WA 98116  
Email: fabiochiussi@gmail.com

Barak Gafni  
Mellanox  
6 Habarzel St.  
Tel Aviv, Israel  
Email: gbarak@mellanox.com

Nabil Bitar  
Alcatel Lucent

Zhenbin Li  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Robert Raszuk  
Email: robert@raszuk.net

Andrew Qu  
MediaTek  
2860 Junction Ave.  
San Jose, CA 95134  
Email: andrew.qu@mediatek.com

MPLS Working Group  
INTERNET-DRAFT  
Updates: 7473 (if approved)  
Intended Status: Proposed Standard  
Expires: December 29, 2017

Santosh Esale  
Raveendra Torvi  
Juniper Networks  
Luay Jalil  
Verizon  
Uma Chunduri  
Huawei  
Kamran Raza  
Cisco Systems, Inc.  
June 27, 2017

Application-aware Targeted LDP  
draft-ietf-mpls-app-aware-tldp-09

Abstract

Recent targeted Label Distribution Protocol (tLDP) applications such as remote loop-free alternate (LFA) and BGP auto discovered pseudowire may automatically establish a tLDP session to any Label Switching Router (LSR) in a network. The initiating LSR has information about the targeted applications to administratively control initiation of the session. However, the responding LSR has no such information to control acceptance of this session. This document defines a mechanism to advertise and negotiate Targeted Applications Capability (TAC) during LDP session initialization. As the responding LSR becomes aware of targeted applications, it may establish a limited number of tLDP sessions for certain applications. In addition, each targeted application is mapped to LDP Forwarding Equivalence Class (FEC) Elements to advertise only necessary LDP FEC-label bindings over the session. This document updates RFC 7473 for enabling advertisement of LDP FEC-label bindings over the session.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

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

## Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1	Introduction	4
1.1	Conventions Used in This Document	4
1.2	Terminology	5
2.	Targeted Application Capability	5
2.1	Encoding	5
2.2	Procedures	6
2.3	LDP message procedures	8
2.3.1	Initialization message	8
2.3.2	Capability message	9
3.	Targeted Application FEC Advertisement Procedures	9
4.	Interaction of Targeted Application Capabilities and State Advertisement Control Capabilities	10
5.	Use cases	12
5.1	Remote LFA Automatic Targeted session	12
5.2	FEC 129 Auto Discovery Targeted session	13
5.3	LDP over RSVP and Remote LFA targeted session	13
5.4	mLDP node protection targeted session	13
6.	Security Considerations	14
7.	IANA Considerations	14
8.	Acknowledgments	15
9.	Contributing Authors	15
10.	References	16

10.1	Normative References . . . . .	16
10.2	Informative References . . . . .	16
	Authors' Addresses . . . . .	17

## 1 Introduction

LDP uses the extended discovery mechanism to establish the tLDP adjacency and subsequent session as described in [RFC5036]. A LSR initiates extended discovery by sending tLDP Hello to specific address. The remote LSR decides to either accept or ignore the tLDP Hello based on local configuration only. Targeted LDP application is an application that uses tLDP session to exchange information such as FEC-Label bindings with a peer LSR in the network. For an application such as FEC 128 pseudowire, the remote LSR is configured with the source LSR address so that it can use that information to accept or ignore given tLDP Hello.

However, applications such as Remote LFA and BGP auto discovered pseudowire automatically initiate asymmetric extended discovery to any LSR in a network based on local state only. With these applications, the remote LSR is not explicitly configured with the source LSR address. So the remote LSR either responds or ignores all tLDP Hellos.

In addition, since the session is initiated and established after adjacency formation, the responding LSR has no targeted applications information available to choose a session with targeted application that it is configured to support. Also, the initiating LSR may employ a limit per application on locally initiated automatic tLDP sessions, however the responding LSR has no such information to employ a similar limit on the incoming tLDP sessions. Further, the responding LSR does not know whether the source LSR is establishing a tLDP session for configured, automatic or both applications.

This document proposes and describes a solution to advertise Targeted Application Capability (TAC), consisting of a targeted application list, during initialization of a tLDP session. It also defines a mechanism to enable a new application and disable an old application after session establishment. This capability advertisement provides the responding LSR with the necessary information to control the acceptance of tLDP sessions per application. For instance, an LSR may accept all BGP auto discovered tLDP sessions as defined in [RFC6074] but may only accept limited number of Remote LFA tLDP sessions as defined in [RFC7490]

Also, the targeted LDP application is mapped to LDP FEC element type to advertise specific application FECs only, avoiding the advertisement of other unnecessary FECs over a tLDP session.

### 1.1 Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",

"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and RFC 8174 [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 1.2 Terminology

In addition to the terminology defined in [RFC7473], this document uses the following terms:

tLDP : Targeted LDP  
 TAC : Targeted Application Capability  
 TAE : Targeted Application Element  
 TA-Id : Targeted Application Identifier  
 SAC : State Advertisement Control Capability  
 LSR : Label Switching Router  
 mLDP : Multipoint LDP  
 PQ : Remote-LFA nexthops  
 RSVP-TE : RSVP Traffic Engineering  
 P2MP : Point-to-Multipoint  
 PW : Pseudowire  
 P2P-PW : Point-to-point Psuedowire  
 MP2MP : Multipoint-to-Multipoint  
 HSMP LSP : Hub and Spoke Multipoint Label Switched Path  
 LSP : Label Switched Path  
 MP2P : Multipoint-to-point  
 MPT : Merge Point

## 2. Targeted Application Capability

### 2.1 Encoding

An LSR MAY advertise that it is capable of negotiating a targeted LDP application list over a tLDP session by using the Capability Advertisement as defined in [RFC5561] and encoded as follows:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-----+-----+-----+-----+-----+-----+-----+-----+
    |U|F| TLV Code Point |                               Length |
    +-----+-----+-----+-----+-----+-----+-----+-----+
    |S| Reserved |                               Capability Data |
    +-----+-----+-----+-----+-----+-----+-----+-----+
    |                                     |
    +-----+-----+-----+-----+-----+-----+-----+
  
```

This document defines a new optional capability TLV of type TBD1 called 'Targeted Application Capability (TAC)'. Flag "U" MUST be set to 1 to indicate that this capability must be silently ignored if unknown. The TAC's Capability Data contains the Targeted Application Element (TAE) information encoded as follows:

Targeted Application Element(TAE)

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|   Targ. Appl. Id                   |E|   Reserved                   |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Targeted Application Identifier (TA-Id):

a 16 bit Targeted Application Identifier value.

E-bit: The enable bit indicates whether the sender is advertising or withdrawing the TAE. The E-bit value is used as follows:

- 1 - The TAE is advertising the targeted application.
- 0 - The TAE is withdrawing the targeted application.

## 2.2 Procedures

At tLDP session establishment time, a LSR MAY include a new capability TLV, TAC TLV, as an optional TLV in the LDP Initialization message. The TAC TLV's Capability data MAY consist of zero or more TAEs each pertaining to a unique TA-Id that a LSR supports over the session. If the receiver LSR receives the same TA-Id in more than one TAE, it MUST process the first element and ignore the duplicate elements. If the receiver LSR receives an unknown TA-Id in the TAE, it MUST silently ignore such a TAE and continue processing the rest of the TLV.

If the receiver LSR does not receive the TAC TLV in the Initialization message or it does not understand the TAC TLV, the TAC negotiation is considered unsuccessful and the session establishment proceeds as per [RFC5036]. On the receipt of a valid TAC TLV, an LSR MUST generate its own TAC TLV with TAEs consisting of unique TA-Ids that it supports over the tLDP session. If there is at least one TAE common between the TAC TLV it has received and its own, the session MUST proceed to establishment as per [RFC5036]. If not, A LSR MUST send a 'Session Rejected/Targeted Application Capability Mis-Match' Notification message to the peer and close the session. The



initiating LSR SHOULD tear down the corresponding tLDP adjacency after sent or receipt of a 'Session Rejected/Targeted Application Capability Mis-Match' Notification message to or from the responding LSR respectively.

If both the peers support TAC TLV, an LSR decides to establish or close a tLDP session based on the negotiated targeted application list. For example, an initiating LSR advertises A, B and C as TA-Ids, and the responding LSR advertises C, D and E as TA-Ids. Then the negotiated TA-Id as per both the LSRs is C. Another example, an initiating LSR advertises A, B and C as TA-Ids, and the responding LSR, which acts as a passive LSR, advertises all the applications - A, B, C, D and E - as TA-Ids that it supports over this session. Then the negotiated targeted applications as per both the LSRs are A, B and C. Finally, If the initiating LSR advertises A, B and C as a TA-Ids and the responding LSR advertises D and E as TA-Ids, then the negotiated targeted applications as per both the LSRs are none. Therefore, if the intersection of the sets of received and sent TA-Id is null, then LSR sends 'Session Rejected/Targeted Application Capability Mis-Match' Notification message to the peer LSR and closes the session.

When the responding LSR playing the active role [RFC5036] in LDP session establishment receives a 'Session Rejected/Targeted Application Capability Mis-Match' Notification message, it MUST set its session setup retry interval to a maximum value, as such 0xffff. The session MAY stay in NON EXISTENT state. When it detects a change in the initiating LSR or local LSR configuration pertaining to TAC TLV, it MUST clear the session setup back off delay associated with the session to re-attempt the session establishment. A LSR detects configuration change on the other LSR with the receipt of tLDP Hello message that has a higher configuration sequence number than the earlier tLDP Hello message.

When the initiating LSR playing the active role in LDP session establishment receives a 'Session Rejected/Targeted Application Capability Mis-Match' Notification message, either it MUST close the session and tear down the corresponding tLDP adjacency or it MUST set its session setup retry interval to a maximum value, as such 0xffff.

If the initiating LSR decides to tear down the associated tLDP adjacency, the session is closed on the initiating as well as the responding LSR. It MAY also take appropriate actions. For instance, if an automatic session intended to support the Remote LFA application is rejected by the responding LSR, the initiating LSR may inform the IGP to calculate another PQ node [RFC7490] for the route or set of routes. More specific actions are a local matter and outside the scope of this document.

If the initiating LSR sets the session setup retry interval to maximum, the session MAY stay in a non-existent state. When this LSR detects a change in the responding LSR configuration or its own configuration pertaining to TAC TLV, it MUST clear the session setup back off delay associated with the session in order to re-attempt the session establishment.

After a tLDP session has been established with TAC capability, the initiating and responding LSR MUST distribute FEC-label bindings for the negotiated applications only. For instance, if the tLDP session is established for BGP auto discovered pseudowire, only FEC 129 label bindings MUST be distributed over the session. Similarly, a LSR operating in downstream on demand mode MUST request FEC-label bindings for the negotiated applications only.

If the Targeted Application Capability and Dynamic Capability, described in [RFC5561], are negotiated during session initialization, TAC MAY be re-negotiated after session establishment by sending an updated TAC TLV in LDP Capability message. The updated TAC TLV carries TA-Ids with incremental update only. The updated TLV MUST consist of one or more TAEs with E-bit set or E-bit off to advertise or withdraw the new and old application respectively. This may lead to advertisements or withdrawals of certain types of FEC-Label bindings over the session or tear down of the tLDP adjacency and subsequently the session.

The Targeted Application Capability is advertised on tLDP session only. If the tLDP session changes to link session, a LSR SHOULD withdraw it with S bit set to 0. Similarly, if the link session changes to tLDP, a LSR SHOULD advertise it via the Capability message. If the capability negotiation fails, this may lead to destruction of the tLDP session.

By default, LSR SHOULD accept tLDP hellos in order to then accept or reject the tLDP session based on the application information.

In addition, LSR SHOULD allow the configuration of any TA-Id in order to facilitate private TA-Id's usage by a network operator.

## 2.3 LDP message procedures

### 2.3.1 Initialization message

1. The S-bit of the Targeted Application Capability TLV MUST be set to 1 to advertise Targeted Application Capability and SHOULD be ignored on the receipt as defined in [RFC5561]
2. The E-bit of the Targeted Application Element MUST be set to 1 to

enable Targeted application and SHOULD be ignored on the receipt.

3. An LSR MAY add State Control Capability by mapping Targeted Application Element to State Advertisement Control (SAC) Elements as defined in Section 4.

### 2.3.2 Capability message

The initiating or responding LSR may re-negotiate the TAC after local configuration change with the Capability message.

1. The S-bit of TAC is set to 1 or 0 to advertise or withdraw it.
2. After configuration change, If there is no common TAE between its new TAE list and peers TAE list, the LSR MUST send a 'Session Rejected/Targeted Application Capability Mis-Match' Notification message and close the session.
3. If there is a common TAE, a LSR MAY also update SAC Capability based on updated TAC as described in section 4 and send the updated TAC and SAC capabilities in a Capability message to the peer.
4. A receiving LSR processes the Capability message with TAC TLV. If the S-bit is set to 0, the TAC is disabled for the session.
5. If the S-bit is set to 1, a LSR process a list of TAEs from TACs capability data with E-bit set to 1 or 0 to update the peer's TAE.

### 3. Targeted Application FEC Advertisement Procedures

The targeted LDP application MUST be mapped to LDP FEC element types as follows to advertise only necessary LDP FEC-Label bindings over the tLDP session.

Targeted Application	Description	FEC mappings
LDPv4 Tunneling	LDP IPv4 over RSVP-TE or other MPLS tunnel	IPv4 prefix
LDPv6 Tunneling	LDP IPv6 over RSVP-TE or other MPLS tunnel	IPv6 prefix
mLDP Tunneling	mLDP over RSVP-TE or other MPLS tunnel	P2MP MP2MP-up

		MP2MP-down HSMP-downstream HSMP-upstream
LDPv4 Remote LFA	LDPv4 over LDPv4 or other MPLS tunnel	IPv4 prefix
LDPv6 Remote LFA	LDPv6 over LDPv6 or other MPLS tunnel	IPv6 prefix
LDP FEC 128 PW	LDP FEC 128 Pseudowire	PWid FEC Element
LDP FEC 129 PW	LDP FEC 129 Pseudowire	Generalized PWid FEC Element
LDP Session Protection	LDP session protection	FEC types as per protected session
LDP ICCP	LDP Inter-chassis control protocol	None
LDP P2MP PW	LDP P2MP Pseudowire	P2MP PW Upstream FEC Element
mLDP Node Protection	mLDP node protection	P2MP MP2MP-up MP2MP-down HSMP-downstream HSMP-upstream
IPv4 intra-area FECs	IPv4 intra-area FECs	IPv4 prefix
IPv6 intra-area FECs	IPv6 intra-area FECs	IPv6 prefix

Intra-area FECs : FECs that are on the shortest path tree and not  
leaves of the shortest path tree.

#### 4. Interaction of Targeted Application Capabilities and State Advertisement Control Capabilities

As described in this document, the set of TAEs negotiated between two LDP peers advertising TAC represents the willingness of both peers to advertise state information for a set of applications. The set of applications negotiated by the TAC mechanism is symmetric between the two LDP peers. In the absence of further mechanisms, two LDP peers will both advertise state information for the same set of applications.

As described in [RFC7473], State Advertisement Control(SAC) TLV can be used by an LDP speaker to communicate its interest or disinterest in receiving state information from a given peer for a particular application. Two LDP peers can use the SAC mechanism to create asymmetric advertisement of state information between the two peers.

The TAC negotiation facilitates the awareness of targeted applications to both the peers. It enables them to advertise only necessary LDP FEC-label bindings corresponding to negotiated applications. With the SAC, the responding LSR is not aware of targeted applications. Thus it may be unable to communicate its interest or disinterest to receive state information from the peer. Therefore, when the responding LSR is not aware of targeted applications such a remote LFA and BGP auto discovered pseudowires, TAC mechanism should be used and when the responding LSR is aware (with appropriate configuration) of targeted applications such as FEC 128 pseudowire, SAC mechanism should be used. Also after TAC mechanism makes the responding LSR aware of targeted application, the SAC mechanism may be used to communicate its disinterest in receiving state information from the peer for a particular negotiated application, creating asymmetric advertisements.

Thus, the TAC mechanism enables two LDP peers to symmetrically advertise state information for negotiated targeted applications. Further, the SAC mechanism enables both of them to asymmetrically disable receipt of state information for some of the already negotiated targeted applications. Collectively, both TAC and SAC mechanisms can be used to control the FEC-label bindings that are advertised over the tLDP session. For instance, suppose the initiating LSR establishes a tLDP session to the responding LSR for Remote LFA and FEC 129 PW targeted applications with TAC. So each LSR advertises the corresponding FEC-Label bindings. Further, suppose the initiating LSR is not the PQ node for responding LSRs Remote LFA IGP calculations. In such a case, the responding LSR may use the SAC mechanism to convey its disinterest in receiving state information for Remote LFA targeted LDP application.

For a given tLDP session, the TAC mechanism can be used without the SAC mechanism, and the SAC mechanism can be used without the TAC mechanism. It is useful to discuss the behavior when TAC and SAC

mechanisms are used on the same tLDP session. The TAC mechanism MUST take precedence over the SAC mechanism with respect to enabling applications for which state information will be advertised. For a tLDP session using the TAC mechanism, the LDP peers MUST NOT advertise state information for an application that has not been negotiated in the most recent TAE list (referred to as an un-negotiated application). This is true even if one of the peers announces its interest in receiving state information that corresponds to the un-negotiated application by sending a SAC TLV. In other words, when TAC is being used, SAC cannot and should not enable state information advertisement for applications that have not been enabled by TAC.

On the other hand, the SAC mechanism MUST take precedence over the TAC mechanism with respect to disabling state information advertisements. If an LDP speaker has announced its disinterest in receiving state information for a given application to a given peer using the SAC mechanism, its peer MUST NOT send state information for that application, even if the two peers have negotiated that the corresponding application via the TAC mechanism.

For the purposes of determining the correspondence between targeted applications defined in this document and application state as defined in [RFC7473] an LSR MUST use the following mappings:

- LDPv4 Tunneling - IPv4 Prefix-LSPs
- LDPv6 Tunneling - IPv6 Prefix-LSPs
- LDPv4 Remote LFA - IPv4 Prefix-LSPs
- LDPv6 Remote LFA - IPv6 Prefix-LSPs
- LDP FEC 128 PW - FEC128 P2P-PW
- LDP FEC 129 PW - FEC129 P2P-PW

An LSR MUST map Targeted Application to LDP capability as follows:

- mLDP Tunneling - P2MP Capability, MP2MP Capability  
and HSMP LSP Capability TLV
- mLDP node protection - P2MP Capability, MP2MP Capability  
and HSMP LSP Capability TLV

## 5. Use cases

### 5.1 Remote LFA Automatic Targeted session

The LSR determines that it needs to form an automatic tLDP session to remote LSR based on IGP calculation as described in [RFC7490] or some other mechanism, which is outside the scope of this document. The LSR

forms the tLDP adjacency and constructs an Initialization message with TAC TLV with TAE as Remote LFA during session establishment. The receiver LSR processes the LDP Initialization message and verifies whether it is configured to accept a Remote LFA tLDP session. If it is, it may further verify that establishing such a session does not exceed the configured limit for Remote LFA sessions. If all these conditions are met, the receiver LSR may respond back with an Initialization message with TAC corresponding to Remote LFA, and subsequently the session may be established.

After the session has been established with TAC capability, the sender and receiver LSR distribute IPv4 or IPv6 FEC label bindings over the session. Further, the receiver LSR may determine that it does not need these FEC label bindings. So it may disable the receipt of these FEC label bindings by mapping targeted application element to state control capability as described in section 4.

## 5.2 FEC 129 Auto Discovery Targeted session

BGP auto discovery may determine whether the LSR needs to initiate an auto-discovery tLDP session with a border LSR. Multiple LSRs may try to form an auto discovered tLDP session with a border LSR. So, a service provider may want to limit the number of auto discovered tLDP sessions a border LSR can accept. As described in Section 2, LDP may convey targeted applications with TAC TLV to border LSR. A border LSR may establish or reject the tLDP session based on local administrative policy. Also, as the receiver LSR becomes aware of targeted applications, it can also employ an administrative policy for security. For instance, it can employ a policy to accept all auto-discovered session from source-list.

Moreover, the sender and receiver LSR must exchange FEC 129 label bindings only over the tLDP session.

## 5.3 LDP over RSVP and Remote LFA targeted session

A LSR may want to establish a tLDP session to a remote LSR for LDP over RSVP tunneling and Remote LFA applications. The sender LSR may add both these applications as a unique Targeted Application Element in the Targeted Application Capability data of a TAC TLV. The receiver LSR may have reached a configured limit for accepting Remote LFA automatic tLDP sessions, but it may have been configured to accept LDP over RSVP tunneling. In such a case, the tLDP session is formed for both LDP over RSVP and Remote LFA applications as both need same FECs - IPv4 or IPv6 or both.

## 5.4 mLDP node protection targeted session

A merge point LSR may determine that it needs to form automatic tLDP session to the upstream point of local repair (PLR) LSR for MP2P and MP2MP LSP [RFC6388] node protection as described in the [RFC7715]. The MPT LSR may add a new targeted LDP application - mLDP protection - as a unique TAE in the Targeted Application Capability Data of a TAC TLV and send it in the Initialization message to the PLR. If the PLR is configured for mLDP node protection and establishing this session does not exceed the limit of either mLDP node protection sessions or automatic tLDP sessions, the PLR may decide to accept this session. Also, the PLR may respond back with the initialization message with a TAC TLV that has one of the TAEs as - mLDP protection, and the session proceeds to establishment as per [RFC5036].

## 6. Security Considerations

The Capability procedure described in this document does not introduce any change to LDP Security Considerations section described in [RFC5036].

As described in [RFC5036], DoS attacks via Extended Hellos, which are required to establish a tLDP session, can be addressed by filtering Extended Hellos using access lists that define addresses with which Extended Discovery is permitted. Further, as described in section 5.2 of this document, a LSR can employ a policy to accept all auto-discovered Extended Hellos from the configured source addresses list.

Also for the two LSRs supporting TAC, the tLDP session is only established after successful negotiation of the TAC. The initiating and receiving LSR MUST only advertise TA-Ids that they support. In other words, what they are configured for over the tLDP session.

## 7. IANA Considerations

This document requires the assignment of a new code point for a Capability Parameter TLVs from the IANA managed LDP registry "TLV Type Name Space", corresponding to the advertisement of the Targeted Applications capability. IANA is requested to assign the lowest available value after 0x050B.

Value	Description	Reference
-----	-----	-----
TBD1	Targeted Applications capability	[this document]

This document requires the assignment of a new code point for a status code from the IANA managed registry "STATUS CODE NAME SPACE" on the Label Distribution Protocol (LDP) Parameters page, corresponding to the notification of session Rejected/Targeted



Application Capability Mis-Match. IANA is requested to assign the lowest available value after 0x0000004B.

Value	E	Description	Reference
TBD2	1	Session Rejected/Targeted Application Capability Mis-Match	[this document]

This document also creates a new name space 'the LDP Targeted Application Identifier' on the Label Distribution Protocol (LDP) Parameters page, that is to be managed by IANA. The range is 0x0001-0xFFFFE, with the following values requested in this document.

Value	Description	Reference
0x0000	Reserved	[this document]
0x0001	LDPv4 Tunneling	[this document]
0x0002	LDPv6 Tunneling	[this document]
0x0003	mLDP Tunneling	[this document]
0x0004	LDPv4 Remote LFA	[this document]
0x0005	LDPv6 Remote LFA	[this document]
0x0006	LDP FEC 128 PW	[this document]
0x0007	LDP FEC 129 PW	[this document]
0x0008	LDP Session Protection	[this document]
0x0009	LDP ICCP	[this document]
0x000A	LDP P2MP PW	[this document]
0x000B	mLDP Node Protection	[this document]
0x000C	LDPv4 Intra-area FECs	[this document]
0x000D	LDPv6 Intra-area FECs	[this document]
0x0001 - 0x1FFF	Available for assignment by IETF Review	
0x2000 - 0F7FF	Available for assignment as first come first served	
0xF800 - 0xFBFF	Available for private use	
0xFC00 - 0xFFFFE	Available for experimental use	
0xFFFF	Reserved	[this document]

## 8. Acknowledgments

The authors wish to thank Nischal Sheth, Hassan Hosseini, Kishore Tiruveedhul, Loa Andersson, Eric Rosen, Yakov Rekhter, Thomas Beckhaus, Tarek Saad, Lizhong Jin and Bruno Decraene for doing the detailed review. Thanks to Manish Gupta and Martin Ehlers for their input to this work and many helpful suggestions.

## 9. Contributing Authors

Chris Bowers  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA 94089  
USA  
EMail: cbowers@juniper.net

Zhenbin Li  
Huawei  
Bld No.156 Beiqing Rd  
Beijing 100095  
China  
Email: lizhenbin@huawei.com

## 10. References

### 10.1 Normative References

- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, October 2007, <<http://www.rfc-editor.org/info/rfc5036>>.
- [RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, July 2009, <<http://www.rfc-editor.org/info/rfc5561>>.
- [RFC7473] Kamran Raza, Sami Boutros, "Controlling State Advertisements of Non-negotiated LDP Applications", RFC 7473, March 2015, <<http://www.rfc-editor.org/info/rfc7473>>.
- [RFC7715] IJ. Wijnands, E. Rosen, K. Raza, J. Tantsura, A. Atlas, Q. Zhao, "mLDP Node Protection", RFC 7715, January 2016, <<http://www.rfc-editor.org/info/rfc7715>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] B. Leiba, "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC8174, May 2017, <<http://www.rfc-editor.org/info/rfc8174>>.

### 10.2 Informative References

- [RFC7490] S. Bryant, C. Filsfils, S. Previdi, M. Shand, N. So,

"Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)",  
April 2015.

[RFC6074] E. Rosen, B. Davie, V. Radoaca, and W. Luo, "Provisioning,  
Auto-Discovery, and Signaling in Layer 2 Virtual Private  
Networks (L2VPNs)", January 2011.

[RFC6388] IJ. Wijnands, I. Minei, K. Kompella, B. Thomas, "Label  
Distribution Protocol Extensions for Point-to-Multipoint  
and Multipoint-to-Multipoint Label Switched Paths",  
November 2011.

#### Authors' Addresses

Santosh Esale  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA 94089  
USA  
EMail: sesale@juniper.net

Raveendra Torvi  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
USA  
EMail: rtorvi@juniper.net

Luay Jalil  
Verizon  
1201 E Arapaho Rd  
Richardson, TX 75081  
USA  
Email: luay.jalil@verizon.com

Uma Chunduri  
Huawei  
2330 Central Expy  
Santa Clara, CA 95050  
USA  
Email: uma.chunduri@huawei.com

Kamran Raza  
Cisco Systems, Inc.  
2000 Innovation Drive  
Ottawa, ON K2K-3E8  
Canada  
E-mail: skraza@cisco.com



TEAS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 13 July 2022

V.P. Beeram  
T. Saad  
Juniper Networks  
R. Gandhi  
Cisco Systems, Inc.  
X. Liu  
Volta Networks  
I. Bryskin  
Individual  
9 January 2022

A YANG Data Model for Resource Reservation Protocol (RSVP)  
draft-ietf-teas-yang-rsvp-17

## Abstract

This document defines a YANG data model for the configuration and management of the RSVP protocol. The YANG data model covers the building blocks that may be augmented by other RSVP extension data models such as RSVP Traffic-Engineering (RSVP-TE). It is divided into two modules that cover the basic and extended RSVP features.

## 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 13 July 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 . . . . .	2
2. Requirements Language . . . . .	3
2.1. Prefixes in Data Node Names . . . . .	3
2.2. Model Tree Diagram . . . . .	4
3. Model Overview . . . . .	4
3.1. Module(s) Relationship . . . . .	5
3.2. Core Features . . . . .	5
3.3. Optional Features . . . . .	6
3.4. Data Model Structure . . . . .	6
3.5. Model Notifications . . . . .	8
4. RSVP Base YANG Model . . . . .	9
4.1. Tree Diagram . . . . .	9
4.2. YANG Module . . . . .	13
5. RSVP Extended YANG Model . . . . .	34
5.1. Tree Diagram . . . . .	34
5.2. YANG Module . . . . .	36
6. IANA Considerations . . . . .	45
7. Security Considerations . . . . .	46
8. Acknowledgement . . . . .	47
9. Appendix A . . . . .	47
10. Contributors . . . . .	53
11. References . . . . .	54
11.1. Normative References . . . . .	54
11.2. Informative References . . . . .	56
Authors' Addresses . . . . .	57

## 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 defines a YANG data model for the configuration and management of the RSVP protocol [RFC2205]. The data model is divided into two modules: a base and extended RSVP YANG modules. The RSVP

base YANG 'ietf-rsvp' module covers the data that is core to the function of the RSVP protocol and MUST be supported by vendors that support RSVP protocol [RFC2205]. The RSVP extended 'ietf-rsvp-extended' module covers the data that is optional, or provides ability to tune RSVP protocol base functionality. The support for RSVP extended module features by vendors is considered optional.

The RSVP YANG model provides the building blocks needed to allow augmentation by other models that extend the RSVP protocol- such as using RSVP extensions to signal Label Switched Paths (LSPs) as defined in [RFC3209].

The YANG module(s) defined in this document are compatible with the Network Management Datastore Architecture (NMDA) [RFC7950].

## 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 terminology for describing YANG data models is found in [RFC7950].

### 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
if	ietf-interfaces	[RFC8343]
rt	ietf-routing	[RFC8349]
rt-types	ietf-routing-types	[RFC8294]
inet	ietf-inet-types	[RFC6991]
yang	ietf-yang-types	[RFC6991]
key-chain	ietf-key-chain	[RFC8177]

Table 1: Prefixes and corresponding YANG modules

## 2.2. Model Tree Diagram

A full tree diagram of the module(s) defined in this document is given in subsequent sections as per the syntax defined in [RFC8340].

## 3. Model Overview

The RSVP YANG module augments the "control-plane-protocol" entry from the 'ietf-routing' module defined in [RFC8349]. It also defines the identity "rsvp" of base type "rt:routing-protocol" to identify the RSVP routing protocol.

The 'ietf-rsvp' model defines a single instance of the RSVP protocol. The top 'rsvp' container encompasses data for one such RSVP protocol instance. Multiple instances can be defined as multiple control-plane protocols instances as described in [RFC8349].

The YANG data model defined has the common building blocks for the operation of the base RSVP protocol for the session type defined in [RFC2205]. The augmentation of this model by other models (e.g. to support RSVP Traffic Engineering (TE) extensions for signaling Label Switched Paths (LSPs)) are outside the scope of this document and are discussed in separate document(s).



### 3.1. Module(s) Relationship

This RSVP YANG data model defined in this document is divided into two modules: a base and extended modules. The RSVP data covered in 'ietf-rsvp' module are categorized as core to the function of the protocol and MUST be supported by vendors claiming the support for RSVP protocol [RFC2205].

The RSVP extended features that are covered in 'ietf-rsvp-extended' module are categorized as either optional or providing ability to better tune the basic functionality of the RSVP protocol. The support for RSVP extended features by all vendors is considered optional.

The relationship between the base and RSVP extended YANG modules and the IETF routing YANG model is shown in Figure 1.

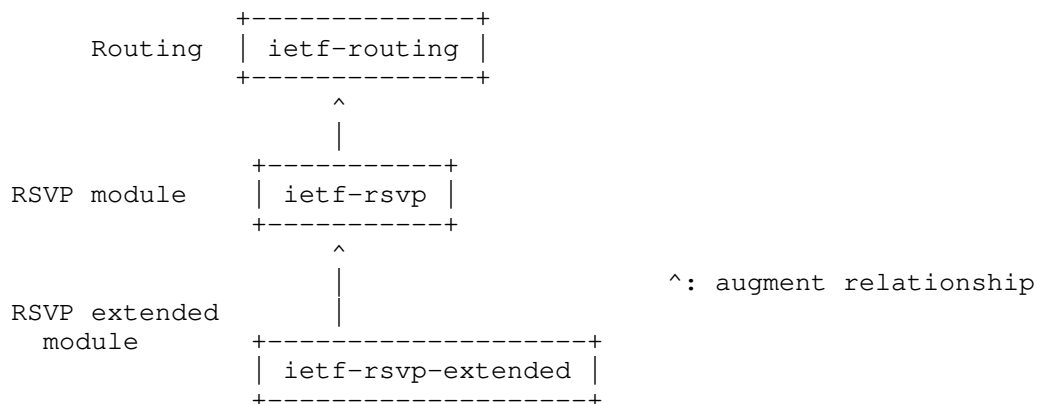


Figure 1: Relationship of RSVP and RSVP extended modules with other protocol modules

### 3.2. Core Features

The RSVP data covered in the 'ietf-rsvp' YANG module provides the common building blocks that are required to configure, operate and manage the RSVP protocol and MUST be supported by vendors that claim the support for base RSVP protocol defined in [RFC2205].

In addition, the following standard RSVP core features are modeled under the 'ietf-rsvp' module:

- \* Basic operational statistics, including protocol messages, packets and errors.

- \* Basic RSVP authentication feature as defined in [RFC2747]) using string based authentication key.
- \* Basic RSVP Refresh Reduction feature as defined in ([RFC2961]).
- \* Basic RSVP Hellos feature as defined in ([RFC3209])
- \* Basic RSVP Graceful Restart feature as defined in [RFC3473], [RFC5063], and [RFC5495].

### 3.3. Optional Features

Optional features are beyond the basic configuration, and operation of the RSVP protocol. The decision whether to support these RSVP features on a particular device is left to the vendor that supports the RSVP core features.

The following optional features that are covered in the 'ietf-rsvp-extended' YANG module:

- \* Advanced operational statistics, including protocol messages, packets and errors.
- \* Advanced RSVP authentication features as defined in [RFC2747]) using various authentication key types including those defined in [RFC8177].
- \* Advanced RSVP Refresh Reduction features defined in ([RFC2961]).
- \* Advanced RSVP Hellos features as defined in [RFC3209], and [rfc4558].
- \* Advanced RSVP Graceful Restart features as defined in [RFC3473], [RFC5063], and [RFC5495].

### 3.4. Data Model Structure

The RSVP YANG data model defines the 'rsvp' top-level container that contains the configuration and operational state for the RSVP protocol. The presence of this container enables the RSVP protocol functionality.

The 'rsvp' top-level container also includes data that has router level scope (i.e. applicable to all objects modeled under rsvp). It also contains configuration and state data about the following types of RSVP objects:

- \* interfaces

\* neighbors

\* sessions

The derived state data is contained in "read-only" nodes directly under the intended object as shown in Figure 2.

```

module: ietf-rsvp
  +--rw rsvp!
    +--rw <<router-level scope data>>
      .
      .
    +--rw interfaces
      .
      +-- ro <<derived state associated with interfaces>>
      .
      .
    +--rw neighbors
      .
      +-- ro <<derived state associated with the LSP Tunnel>>
      .
      .
    +--rw sessions
      .
      +-- ro <<derived state associated with the LSP Tunnel>>
      .
  rpcs:
    +--x clear-session
    +--x clear-neighbor
    +--x clear-authentication

```

Figure 2: RSVP high-level tree model view

The following

'router-level':

The router-level scope configuration and state data are applicable to all modeled objects under the top-level 'rsvp' container, and MAY affect the RSVP protocol behavior.

'interfaces':

The 'interfaces' container includes a list of RSVP enabled interfaces. It also includes RSVP configuration and state data that is applicable to all interfaces. An entry in the interfaces list MAY carry its own configuration or state data. Any data or state under the "interfaces" container level is equally applicable to all interfaces unless it is explicitly overridden by configuration or state under a specific interface.

'neighbors' :

The 'neighbors' container includes a list of RSVP neighbors. An entry in the RSVP neighbor list MAY carry its own configuration and state relevant to the specific RSVP neighbor. The RSVP neighbors can be dynamically discovered using RSVP signaling, or can be explicitly configured.

'sessions' :

The 'sessions' container includes a list RSVP sessions. An entry in the RSVP session list MAY carry its own configuration and state relevant to a specific RSVP session. RSVP sessions are usually derived state that are created as result of signaling. This model defines attributes related to IP RSVP sessions as defined in [RFC2205].

The defined YANG data model supports configuration inheritance for neighbors, and interfaces. Data nodes defined under the main container (e.g. the container that encompasses the list of interfaces, or neighbors) are assumed to apply equally to all elements of the list, unless overridden explicitly for a certain element (e.g. interface).

### 3.5. Model Notifications

Modeling notifications data is key in any defined YANG data model. [RFC8639] and [RFC8641] define a subscription and push mechanism for YANG datastores. This mechanism currently allows the user to:

- \* Subscribe notifications on a per client basis
- \* Specify subtree filters [RFC6241] or XPath filters [RFC8639] so that only interested contents will be sent.
- \* Specify either periodic or on-demand notifications.

#### 4. RSVP Base YANG Model

The RSVP base module includes the core features and building blocks for modeling the RSVP protocol as described in Section 3.2.

##### 4.1. Tree Diagram

Figure 3 shows the YANG tree representation for configuration, state data and RPCs that are covered in 'ietf-rsvp' YANG module:

```

module: ietf-rsvp

augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol:
    +--rw rsvp!
      +--rw interfaces
        +--rw refresh-reduction
          | +--rw enabled?    boolean
        +--rw hellos
          | +--rw enabled?    boolean
        +--rw authentication
          | +--rw enabled?      boolean
          | +--rw authentication-key? string
          | +--rw crypto-algorithm identityref
        +--ro statistics
          +--ro messages
            +--ro ack-sent?          yang:counter64
            +--ro ack-received?      yang:counter64
            +--ro bundle-sent?       yang:counter64
            +--ro bundle-received?   yang:counter64
            +--ro hello-sent?        yang:counter64
            +--ro hello-received?    yang:counter64
            +--ro integrity-challenge-sent? yang:counter64
            +--ro integrity-challenge-received? yang:counter64
            +--ro integrity-response-sent? yang:counter64
            +--ro integrity-response-received? yang:counter64
            +--ro notify-sent?        yang:counter64
            +--ro notify-received?    yang:counter64
            +--ro path-sent?          yang:counter64
            +--ro path-received?      yang:counter64
            +--ro path-err-sent?      yang:counter64
            +--ro path-err-received?  yang:counter64
            +--ro path-tear-sent?     yang:counter64
            +--ro path-tear-received? yang:counter64
            +--ro resv-sent?          yang:counter64
            +--ro resv-received?      yang:counter64
            +--ro resv-confirm-sent?  yang:counter64
            +--ro resv-confirm-received? yang:counter64

```

```

+--ro resv-err-sent?                yang:counter64
+--ro resv-err-received?            yang:counter64
+--ro resv-tear-sent?               yang:counter64
+--ro resv-tear-received?           yang:counter64
+--ro srefresh-sent?                yang:counter64
+--ro srefresh-received?            yang:counter64
+--ro unknown-messages-received?    yang:counter64
+--ro packets
|   +--ro sent?                     yang:counter64
|   +--ro received?                 yang:counter64
+--ro errors
|   +--ro authenticate?             yang:counter64
|   +--ro checksum?                 yang:counter64
|   +--ro packet-length?            yang:counter64
+--rw interface* [name]
|   +--rw name                       if:interface-ref
|   +--rw refresh-reduction
|   |   +--rw enabled?              boolean
|   +--rw hellos
|   |   +--rw enabled?              boolean
|   +--rw authentication
|   |   +--rw enabled?              boolean
|   |   +--rw authentication-key?   string
|   |   +--rw crypto-algorithm      identityref
+--ro statistics
|   +--ro messages
|   |   +--ro ack-sent?
|   |   |   yang:counter64
|   |   +--ro ack-received?
|   |   |   yang:counter64
|   |   +--ro bundle-sent?
|   |   |   yang:counter64
|   |   +--ro bundle-received?
|   |   |   yang:counter64
|   |   +--ro hello-sent?
|   |   |   yang:counter64
|   |   +--ro hello-received?
|   |   |   yang:counter64
|   |   +--ro integrity-challenge-sent?
|   |   |   yang:counter64
|   |   +--ro integrity-challenge-received?
|   |   |   yang:counter64
|   |   +--ro integrity-response-sent?
|   |   |   yang:counter64
|   |   +--ro integrity-response-received?
|   |   |   yang:counter64
|   |   +--ro notify-sent?
|   |   |   yang:counter64

```

```

+--ro notify-received?
|   yang:counter64
+--ro path-sent?
|   yang:counter64
+--ro path-received?
|   yang:counter64
+--ro path-err-sent?
|   yang:counter64
+--ro path-err-received?
|   yang:counter64
+--ro path-tear-sent?
|   yang:counter64
+--ro path-tear-received?
|   yang:counter64
+--ro resv-sent?
|   yang:counter64
+--ro resv-received?
|   yang:counter64
+--ro resv-confirm-sent?
|   yang:counter64
+--ro resv-confirm-received?
|   yang:counter64
+--ro resv-err-sent?
|   yang:counter64
+--ro resv-err-received?
|   yang:counter64
+--ro resv-tear-sent?
|   yang:counter64
+--ro resv-tear-received?
|   yang:counter64
+--ro srefresh-sent?
|   yang:counter64
+--ro srefresh-received?
|   yang:counter64
+--ro unknown-messages-received?
|   yang:counter64
+--ro packets
|   +--ro sent?          yang:counter64
|   +--ro received?     yang:counter64
+--ro errors
|   +--ro authenticate?  yang:counter64
|   +--ro checksum?      yang:counter64
|   +--ro packet-length? yang:counter64
+--rw sessions
|   +--ro session-ip*
|       [destination protocol-id destination-port]
|       +--ro destination-port  uint16
|       +--ro protocol-id       uint8

```

```

    +--ro source?                inet:ip-address
    +--ro destination            inet:ip-address
    +--ro session-name?          string
    +--ro session-status?        enumeration
    +--ro session-type           identityref
    +--ro psbs
      +--ro psb* []
        +--ro source-port?      inet:port-number
        +--ro expires-in?       uint32
    +--ro rsbs
      +--ro rsb* []
        +--ro source-port?      inet:port-number
        +--ro reservation-style identityref
        +--ro expires-in?       uint32
+--rw neighbors
  +--rw neighbor* [address]
    +--rw address                inet:ip-address
    +--rw epoch?                 uint32
    +--rw expiry-time?           uint32
    +--rw graceful-restart
      +--ro neighbor-restart-time? uint32
      +--ro neighbor-recovery-time? uint32
      +--rw helper-mode
        +--ro neighbor-restart-time-remaining? uint32
        +--ro neighbor-recovery-time-remaining? uint32
    +--ro hello-status?          enumeration
    +--rw interface?             if:interface-ref
    +--ro neighbor-status?       enumeration
    +--rw refresh-reduction-capable? boolean
    +--ro restart-count?         yang:counter32
    +--ro restart-time?          yang:date-and-time
+--rw graceful-restart
  +--rw enabled?                 boolean
  +--rw local-restart-time?      uint32
  +--rw local-recovery-time?     uint32
  +--rw helper-mode
    +--rw enabled?               boolean
    +--rw max-helper-restart-time? uint32
    +--rw max-helper-recovery-time? uint32

rpcs:
  +---x clear-session
    +---w input
      +---w routing-protocol-instance-name leafref
      +---w (filter-type)
        +---:(match-all)
          +---w all                  empty
        +---:(match-one)

```



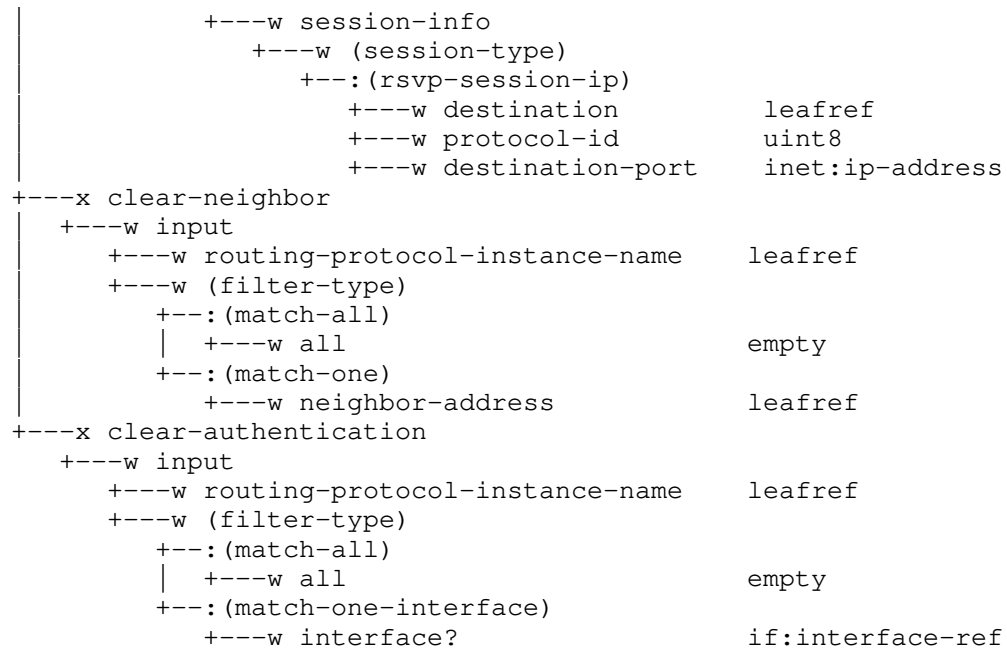


Figure 3: RSVP model tree diagram

#### 4.2. YANG Module

The ietf-rsvp module imports from the following modules:

- \* ietf-interfaces defined in [RFC8343]
- \* ietf-yang-types and ietf-inet-types defined in [RFC6991]
- \* ietf-routing defined in [RFC8349]
- \* ietf-key-chain defined in [RFC8177]
- \* ietf-netconf-acm defined in [RFC8341]

This module also references the following documents: [RFC2205], [RFC5495], [RFC3473], [RFC5063], [RFC2747], [RFC3209], and [RFC2961].

```

<CODE BEGINS> file "ietf-rsvp@2021-12-02.yang"
module ietf-rsvp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp";

  /* Replace with IANA when assigned */

```

```
prefix rsvp;

import ietf-interfaces {
  prefix if;
  reference
    "RFC8343: A YANG Data Model for Interface Management";
}
import ietf-inet-types {
  prefix inet;
  reference
    "RFC6991: Common YANG Data Types";
}
import ietf-yang-types {
  prefix yang;
  reference
    "RFC6991: Common YANG Data Types";
}
import ietf-routing {
  prefix rt;
  reference
    "RFC8349: A YANG Data Model for Routing Management
    (NMDA Version)";
}
import ietf-key-chain {
  prefix key-chain;
  reference
    "RFC8177: YANG Data Model for Key Chains";
}
import ietf-netconf-acm {
  prefix nacm;
  reference
    "RFC8341: Network Configuration Access Control Model";
}
organization
  "IETF Traffic Engineering Architecture and Signaling (TEAS)
  Working Group";
contact
  "WG Web:    <http://tools.ietf.org/wg/teas/>
  WG List:    <mailto:teas@ietf.org>

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

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

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

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

Editor: Igor Bryskin  
<mailto:i\_bryskin@yahoo.com>;

description

"This module contains the RSVP YANG data model.  
The model fully conforms to the Network Management Datastore  
Architecture (NMDA)."

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

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

// RFC Ed.: replace XXXX with actual RFC number and remove this  
// note.  
// RFC Ed.: update the date below with the date of RFC publication  
// and remove this note.

```
revision 2021-12-02 {  
  description  
    "Initial version.";  
  reference  
    "RFCXXXX: A YANG Data Model for Resource Reservation Protocol  
    (RSVP)";  
}
```

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

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

```
identity rsvp-session-ip {  
  base rsvp-session-type;
```

```
    description
      "RSVP IP session type";
  }

  identity reservation-style {
    description
      "Base identity for reservation style.";
  }

  identity reservation-wildcard-filter {
    base reservation-style;
    description
      "Wildcard-Filter (WF) Style.";
    reference
      "RFC2205";
  }

  identity reservation-fixed-filter {
    base reservation-style;
    description
      "Fixed-Filter (FF) Style.";
    reference
      "RFC2205";
  }

  identity reservation-shared-explicit {
    base reservation-style;
    description
      "Shared Explicit (SE) Style.";
    reference
      "RFC2205";
  }

  grouping graceful-restart {
    description
      "RSVP graceful restart local parameters grouping.";
    container graceful-restart {
      description
        "Graceful restart local information.";
      leaf enabled {
        type boolean;
        default "false";
        description
          "'true' if RSVP Graceful Restart is enabled.
          'false' if RSVP Graceful Restart is disabled.";
        reference "RFC5495";
      }
      leaf local-restart-time {
```

```
    type uint32;
    units "seconds";
    default "120";
    description
        "Time it takes the local node to restart its RSVP-TE
        component (to the point where it can exchange RSVP
        Hello with its neighbors). A value of 0xffffffff
        indicates that the restart of the neighbor's control plane
        may occur over an indeterminate interval and that the
        operation of its data plane is unaffected by control plane
        failures.";
    reference "RFC3473";
}
leaf local-recovery-time {
    type uint32;
    units "seconds";
    default "120";
    description
        "The period of time, in seconds, that the local
        node requires to re-synchronize RSVP and MPLS
        forwarding state with its neighbor. A value of zero (0)
        indicates that MPLS forwarding state was not preserved
        across a particular reboot.";
    reference "RFC3473";
}
container helper-mode {
    description
        "Helper mode information. In this mode, the node
        resynchronize its stored states with a neighbor whose
        control plane has restarted. The helper mode term is
        borrowed from RFC3623 and adopted by several vendors
        vendors in their implementation of RSVP graceful restart.";
    leaf enabled {
        type boolean;
        default "true";
        description
            "'true' if helper mode is enabled.";
    }
    leaf max-helper-restart-time {
        type uint32;
        units "seconds";
        default "20";
        description
            "The maximum time the router or switch waits after it
            discovers that the neighboring router has gone down
            before it declares the neighbor down.";
        reference "RFC5063";
    }
}
```

```
    leaf max-helper-recovery-time {
      type uint32;
      units "seconds";
      default "180";
      description
        "The maximum amount of time the router retains the state
         of its RSVP neighbors while they undergo a graceful
         restart.";
      reference "RFC5063";
    }
  }
}

grouping neighbor-graceful-restart {
  description
    "RSVP graceful restart neighbor parameters grouping.";
  container graceful-restart {
    description
      "Graceful restart information.";
    leaf neighbor-restart-time {
      type uint32;
      units "seconds";
      default "120";
      config false;
      description
        "Time it takes the neighbor node to restart its RSVP-TE
         component (to the point where it can exchange RSVP
         Hello with its neighbors). A value of 0xffffffff
         indicates that the restart of the neighbor's control plane
         may occur over an indeterminate interval and that the
         operation of its data plane is unaffected by control plane
         failures.";
      reference "RFC3473";
    }
    leaf neighbor-recovery-time {
      type uint32;
      units "seconds";
      default "120";
      config false;
      description
        "The period of time, in milliseconds, that the neighbor
         node requires to re-synchronize RSVP and MPLS
         forwarding state with its neighbor. A value of zero (0)
         indicates that MPLS forwarding state was not preserved
         across a particular reboot.";
      reference "RFC3473";
    }
  }
  container helper-mode {
```

```
    description
      "Helper mode information.";
    leaf neighbor-restart-time-remaining {
      type uint32;
      units "seconds";
      config false;
      description
        "Number of seconds remaining for neighbor to send Hello
        message after restart.";
      reference "RFC5063";
    }
    leaf neighbor-recovery-time-remaining {
      type uint32;
      units "seconds";
      config false;
      description
        "Number of seconds remaining for neighbor to refresh.";
      reference "RFC5063";
    }
  }
  // helper-mode
}

grouping refresh-reduction {
  description
    "Top level grouping for RSVP refresh reduction parameters.";
  container refresh-reduction {
    description
      "Top level container for RSVP refresh reduction parameters.";
    leaf enabled {
      type boolean;
      default "true";
      description
        "'true' if RSVP Refresh Reduction is enabled.
        'false' if RSVP Refresh Reduction is disabled.";
    }
    reference
      "RFC2961 RSVP Refresh Overhead Reduction Extensions";
  }
}

grouping authentication {
  description
    "Top level grouping for RSVP authentication parameters.";
  container authentication {
    description
      "Top level container for RSVP authentication parameters.";
  }
}
```

```
    leaf enabled {
      type boolean;
      default "false";
      description
        "'true' if RSVP Authentication is enabled.
        'false' if RSVP Authentication is disabled.";
    }
    leaf authentication-key {
      type string;
      default "";
      description
        "An authentication key string.";
      reference
        "RFC2747: RSVP Cryptographic Authentication";
    }
    leaf crypto-algorithm {
      type identityref {
        base key-chain:crypto-algorithm;
      }
      mandatory true;
      description
        "Cryptographic algorithm associated with key.";
    }
  }
}

grouping hellos {
  description
    "Top level grouping for RSVP hellos parameters.";
  container hellos {
    description
      "Top level container for RSVP hello parameters.";
    leaf enabled {
      type boolean;
      default "true";
      description
        "'true' if RSVP Hello is enabled.
        'false' if RSVP Hello is disabled.";
    }
    reference
      "RFC3209: RSVP-TE: Extensions to RSVP for LSP Tunnels.
      RFC5495: Description of the Resource Reservation Protocol -
      Traffic-Engineered (RSVP-TE) Graceful Restart Procedures.";
  }
}

grouping session-attributes {
  description
```



```
    "Top level grouping for RSVP session properties.";
  leaf destination-port {
    type uint16;
    description
      "RSVP destination port.";
    reference
      "RFC2205";
  }
  leaf protocol-id {
    type uint8;
    description
      "The IP protocol ID.";
    reference
      "RFC2205, section 3.2";
  }
  leaf source {
    type inet:ip-address;
    description
      "RSVP source address.";
    reference
      "RFC2205";
  }
  leaf destination {
    type inet:ip-address;
    description
      "RSVP destination address.";
    reference
      "RFC2205";
  }
  leaf session-name {
    type string;
    default "";
    description
      "The signaled name of this RSVP session.";
  }
  leaf session-status {
    type enumeration {
      enum up {
        description
          "RSVP session is up.";
      }
      enum down {
        description
          "RSVP session is down.";
      }
    }
    default "down";
    description
```

```
        "Enumeration of RSVP session states.";
    }
    leaf session-type {
        type identityref {
            base rsvp-session-type;
        }
        mandatory "true";
        description
            "RSVP session type.";
    }
    container psbs {
        description
            "Path State Block (PSB) container.";
        list psb {
            description
                "List of Path State Blocks.";
            leaf source-port {
                type inet:port-number;
                description
                    "RSVP source port.";
                reference
                    "RFC2205";
            }
            leaf expires-in {
                type uint32;
                units "seconds";
                default "180";
                description
                    "Time to expiry (in seconds).";
            }
        }
    }
    container rsbs {
        description
            "Reservation State Block (RSB) container.";
        list rsb {
            description
                "List of Reservation State Blocks.";
            leaf source-port {
                type inet:port-number;
                description
                    "RSVP source port.";
                reference
                    "RFC2205";
            }
            leaf reservation-style {
                type identityref {
                    base reservation-style;
                }
            }
        }
    }
}
```

```
    }
    mandatory "true";
    description
      "RSVP reservation style.";
  }
  leaf expires-in {
    type uint32;
    units "seconds";
    default "180";
    description
      "Time to expiry (in seconds).";
  }
}

grouping neighbor-attributes {
  description
    "Top level grouping for RSVP neighbor properties.";
  leaf address {
    type inet:ip-address;
    description
      "Address of the RSVP neighbor.";
  }
  leaf epoch {
    type uint32;
    default "0";
    description
      "Neighbor epoch.";
    reference "RFC5063";
  }
  leaf expiry-time {
    type uint32;
    units "seconds";
    default "180";
    description
      "Neighbor expiry time after which the neighbor state is
      purged if no states associated with it.";
  }
  uses neighbor-graceful-restart {
    description
      "Allows configuration applicable to all
      neighbors";
  }
  leaf hello-status {
    type enumeration {
      enum enabled {
        description
```

```
        "RSVP Hellos enabled.";
    }
    enum disabled {
        description
            "RSVP Hellos disabled.";
    }
    enum restarting {
        description
            "RSVP restarting.";
    }
}
config false;
description
    "RSVP Hello status.";
}
leaf interface {
    type if:interface-ref;
    description
        "Interface where RSVP neighbor was detected.";
}
leaf neighbor-status {
    type enumeration {
        enum up {
            description
                "Neighbor state up.";
        }
        enum down {
            description
                "Neighbor state down.";
        }
        enum hello-disable {
            description
                "RSVP Hellos disabled.";
        }
        enum restarting {
            description
                "RSVP neighbor restarting.";
        }
    }
}
config false;
description
    "RSVP neighbor state.";
}
leaf refresh-reduction-capable {
    type boolean;
    default "true";
    description
        "Enables all RSVP refresh reduction message bundling, RSVP
```

```
        message ID, reliable message delivery and Srefresh
        messages.";
    reference
        "RFC2961 RSVP Refresh Overhead Reduction Extensions";
}
leaf restart-count {
    type yang:counter32;
    config false;
    description
        "Number of times this RSVP neighbor has restarted.";
}
leaf restart-time {
    type yang:date-and-time;
    config false;
    description
        "Last restart time of the RSVP neighbor.";
    reference "RFC3473";
}
}

grouping packet-statistics {
    description
        "Packet statistics grouping.";
    container packets {
        description
            "Packet statistics container.";
        leaf sent {
            type yang:counter64;
            description
                "RSVP packet sent count.";
        }
        leaf received {
            type yang:counter64;
            description
                "RSVP packet received count.";
        }
    }
}

grouping message-statistics {
    description
        "RSVP protocol statistics grouping.";
    container messages {
        description
            "RSVP protocol statistics container.";
        leaf ack-sent {
            type yang:counter64;
            description
```

```
        "RSVP Hello sent count.";
    }
    leaf ack-received {
        type yang:counter64;
        description
            "RSVP Hello received count.";
    }
    leaf bundle-sent {
        type yang:counter64;
        description
            "RSVP Bundle message sent count.";
    }
    leaf bundle-received {
        type yang:counter64;
        description
            "RSVP Bundle message received count.";
    }
    leaf hello-sent {
        type yang:counter64;
        description
            "RSVP Hello message sent count.";
    }
    leaf hello-received {
        type yang:counter64;
        description
            "RSVP Hello message received count.";
    }
    leaf integrity-challenge-sent {
        type yang:counter64;
        description
            "RSVP Integrity Challenge message sent count.";
    }
    leaf integrity-challenge-received {
        type yang:counter64;
        description
            "RSVP Integrity Challenge message received count.";
    }
    leaf integrity-response-sent {
        type yang:counter64;
        description
            "RSVP Integrity Response message sent count.";
    }
    leaf integrity-response-received {
        type yang:counter64;
        description
            "RSVP Integrity Response message received count.";
    }
    leaf notify-sent {
```

```
    type yang:counter64;
    description
      "RSVP Notify message sent count.";
  }
  leaf notify-received {
    type yang:counter64;
    description
      "RSVP Notify message received count.";
  }
  leaf path-sent {
    type yang:counter64;
    description
      "RSVP Path message sent count.";
  }
  leaf path-received {
    type yang:counter64;
    description
      "RSVP Path message received count.";
  }
  leaf path-err-sent {
    type yang:counter64;
    description
      "RSVP Path error message sent count.";
  }
  leaf path-err-received {
    type yang:counter64;
    description
      "RSVP Path error message received count.";
  }
  leaf path-tear-sent {
    type yang:counter64;
    description
      "RSVP Path tear message sent count.";
  }
  leaf path-tear-received {
    type yang:counter64;
    description
      "RSVP Path tear message received count.";
  }
  leaf resv-sent {
    type yang:counter64;
    description
      "RSVP Resv message sent count.";
  }
  leaf resv-received {
    type yang:counter64;
    description
      "RSVP Resv message received count.";
```

```
    }
    leaf resv-confirm-sent {
      type yang:counter64;
      description
        "RSVP Confirm message sent count.";
    }
    leaf resv-confirm-received {
      type yang:counter64;
      description
        "RSVP Confirm message received count.";
    }
    leaf resv-err-sent {
      type yang:counter64;
      description
        "RSVP Resv error message sent count.";
    }
    leaf resv-err-received {
      type yang:counter64;
      description
        "RSVP Resv error message received count.";
    }
    leaf resv-tear-sent {
      type yang:counter64;
      description
        "RSVP Resv tear message sent count.";
    }
    leaf resv-tear-received {
      type yang:counter64;
      description
        "RSVP Resv tear message received count.";
    }
    leaf srefresh-sent {
      type yang:counter64;
      description
        "RSVP Srefresh message sent count.";
    }
    leaf srefresh-received {
      type yang:counter64;
      description
        "RSVP Srefresh message received count.";
    }
    leaf unknown-messages-received {
      type yang:counter64;
      description
        "Unknown messages received count.";
    }
  }
}
```



```
grouping errors-statistics {
  description
    "Error statistics grouping.";
  container errors {
    description
      "Error statistics container.";
    leaf authenticate {
      type yang:counter64;
      description
        "The total number of RSVP packets received with an
        authentication failure.";
    }
    leaf checksum {
      type yang:counter64;
      description
        "The total number of RSVP packets received with an invalid
        checksum value.";
    }
    leaf packet-length {
      type yang:counter64;
      description
        "The total number of packets received with an invalid
        packet length.";
    }
  }
}

grouping statistics {
  description
    "RSVP statistic attributes.";
  container statistics {
    config false;
    description
      "RSVP statistics container.";
    uses message-statistics;
    uses packet-statistics;
    uses errors-statistics;
  }
}

grouping intf-attributes {
  description
    "Top level grouping for RSVP interface properties.";
  uses refresh-reduction;
  uses hellos;
  uses authentication;
  uses statistics;
}
```

```
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol" {
    when "rt:type = 'rsvp:rsvp'" {
      description
        "This augment is only valid when routing protocol instance
        type is RSVP.";
    }
    description
      "RSVP protocol augmentation.";
    container rsvp {
      presence "Enable RSVP feature";
      description
        "RSVP feature container";
      container interfaces {
        description
          "RSVP interfaces container.";
        uses intf-attributes;
        list interface {
          key "name";
          description
            "RSVP interfaces.";
          leaf name {
            type if:interface-ref;
            description
              "RSVP interface.";
          }
          uses intf-attributes;
        }
      }
    }
    container sessions {
      description
        "RSVP sessions container.";
      list session-ip {
        key "destination protocol-id destination-port";
        config false;
        description
          "List of RSVP sessions.";
        uses session-attributes;
      }
    }
    container neighbors {
      description
        "RSVP neighbors container";
      list neighbor {
        key "address";
        description
          "List of RSVP neighbors";
        uses neighbor-attributes;
      }
    }
  }
```

```
    }
  }
  uses graceful-restart;
}

grouping session-ref {
  description
    "Session reference information";
  leaf destination {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols"
        + "/rt:control-plane-protocol/rsvp:rsvp"
        + "/rsvp:sessions/rsvp:session-ip/destination";
    }
    mandatory true;
    description
      "The RSVP session destination.";
  }
  leaf protocol-id {
    type uint8;
    mandatory true;
    description
      "The RSVP session protocol ID.";
  }
  leaf destination-port {
    type inet:ip-address;
    mandatory true;
    description
      "The RSVP session destination port.";
  }
}

rpc clear-session {
  nacm:default-deny-all;
  description
    "Clears RSVP sessions RPC";
  input {
    leaf routing-protocol-instance-name {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/"
          + "rt:control-plane-protocol/rt:name";
      }
      mandatory true;
      description
        "Name of the RSVP protocol instance whose session
        is being cleared."
    }
  }
}
```

```
        If the corresponding RSVP instance doesn't exist,
        then the operation will fail with an error-tag of
        'data-missing' and an error-app-tag of
        'routing-protocol-instance-not-found'.";
    }
    choice filter-type {
        mandatory true;
        description
            "Filter choice";
        case match-all {
            leaf all {
                type empty;
                mandatory true;
                description
                    "Match all RSVP sessions.";
            }
        }
        case match-one {
            container session-info {
                description
                    "Specifies the specific session to invoke the operation
                    on.";
                choice session-type {
                    mandatory true;
                    description
                        "The RSVP session type.";
                    case rsvp-session-ip {
                        uses session-ref;
                    }
                }
            }
        }
    }
}

rpc clear-neighbor {
    nacm:default-deny-all;
    description
        "RPC to clear the RSVP Hello session to a neighbor.";
    input {
        leaf routing-protocol-instance-name {
            type leafref {
                path "/rt:routing/rt:control-plane-protocols/"
                    + "rt:control-plane-protocol/rt:name";
            }
            mandatory true;
            description

```

```
        "Name of the RSVP protocol instance whose session
        is being cleared.

        If the corresponding RSVP instance doesn't exist,
        then the operation will fail with an error-tag of
        'data-missing' and an error-app-tag of
        'routing-protocol-instance-not-found'.";
    }
    choice filter-type {
        mandatory true;
        description
            "The Filter choice.";
        case match-all {
            leaf all {
                type empty;
                mandatory true;
                description
                    "Match all RSVP neighbor sessions.";
            }
        }
        case match-one {
            leaf neighbor-address {
                type leafref {
                    path "/rt:routing/rt:control-plane-protocols"
                        + "/rt:control-plane-protocol/rsvp:rsvp"
                        + "/rsvp:neighbors/rsvp:neighbor/address";
                }
                mandatory true;
                description
                    "Match the specific RSVP neighbor session.";
            }
        }
    }
}

rpc clear-authentication {
    nacm:default-deny-all;
    description
        "Clears the RSVP Security Association (SA) before the
        lifetime expires.";
    input {
        leaf routing-protocol-instance-name {
            type leafref {
                path "/rt:routing/rt:control-plane-protocols/"
                    + "rt:control-plane-protocol/rt:name";
            }
            mandatory true;
        }
    }
}
```



```

module: ietf-rsvp-extended

augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp
    /rsvp:graceful-restart:
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces:
    +---rw refresh-interval?          uint32
    +---rw refresh-misses?            uint32
    +---rw checksum-enable?           empty
    +---rw patherr-state-removal?      empty
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:statistics/rsvp:packets:
    +---ro discontinuity-time?         yang:date-and-time
    +---ro out-dropped?                yang:counter64
    +---ro in-dropped?                yang:counter64
    +---ro out-errors?                yang:counter64
    +---ro in-errors?                 yang:counter64
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:refresh-reduction:
    +---rw bundle-message-max-size?   uint32
    +---rw ack-hold-time?              uint32
    +---rw ack-max-size?              uint32
    +---rw ack-retransmit-time?       uint32
    +---rw srefresh-ack-desired?       empty
    +---rw srefresh-max-size?         uint32
    +---rw srefresh-relative-period?  uint8
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:hellos:
    +---rw interface-based?           empty
    +---rw hello-interval?            uint32
    +---rw hello-misses?              uint32
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:authentication:
    +---rw lifetime?                  uint32
    +---rw window-size?               uint32
    +---rw challenge?                 empty
    +---rw retransmits?               uint32
    +---rw key-chain?                 key-chain:key-chain-ref
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:interface:
    +---rw refresh-interval?          uint32
    +---rw refresh-misses?            uint32

```

```

    +--rw checksum-enable?          empty
    +--rw patherr-state-removal?    empty
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:interface/rsvp:statistics/rsvp:packets:
    +--ro discontinuity-time?      yang:date-and-time
    +--ro out-dropped?             yang:counter64
    +--ro in-dropped?             yang:counter64
    +--ro out-errors?             yang:counter64
    +--ro in-errors?              yang:counter64
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:interface/rsvp:refresh-reduction:
    +--rw bundle-message-max-size? uint32
    +--rw ack-hold-time?           uint32
    +--rw ack-max-size?           uint32
    +--rw ack-retransmit-time?    uint32
    +--rw srefresh-ack-desired?   empty
    +--rw srefresh-max-size?      uint32
    +--rw srefresh-relative-period? uint8
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:interface/rsvp:hellos:
    +--rw interface-based?        empty
    +--rw hello-interval?         uint32
    +--rw hello-misses?          uint32
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
    /rsvp:interface/rsvp:authentication:
    +--rw lifetime?              uint32
    +--rw window-size?          uint32
    +--rw challenge?             empty
    +--rw retransmits?           uint32
    +--rw key-chain?             key-chain:key-chain-ref

```

Figure 4: RSVP extended module tree diagram

## 5.2. YANG Module

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

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



Figure 5 shows the RSVP extended YANG module:

This module also references the following documents: [RFC3473], [RFC2747], [RFC3209], [RFC2205], [RFC2961], and [RFC5495].

```
<CODE BEGINS> file "ietf-rsvp-extended@2021-12-02.yang"
module ietf-rsvp-extended {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp-extended";
  prefix rsvp-extended;

  import ietf-rsvp {
    prefix rsvp;
    reference
      "RFCXXXX: A YANG Data Model for Resource Reservation Protocol
      (RSVP)";
  }
  import ietf-routing {
    prefix rt;
    reference
      "RFC8349: A YANG Data Model for Routing Management
      (NMDA Version)";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC6991: Common YANG Data Types";
  }
  import ietf-key-chain {
    prefix key-chain;
    reference
      "RFC8177: YANG Data Model for Key Chains";
  }

  organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS)
    Working Group";
  contact
    "WG Web:  <http://tools.ietf.org/wg/teas/>
    WG List:  <mailto:teas@ietf.org>

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

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

    Editor:   Rakesh Gandhi
```

```
<mailto:rgandhi@cisco.com>

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

Editor:  Igor Bryskin
        <mailto:i_bryskin@yahoo.com>";

description
  "This module contains the Extended RSVP YANG data model.
  The model fully conforms to the Network Management Datastore
  Architecture (NMDA).

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

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

// 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 2021-12-02 {
  description
    "Initial version.";
  reference
    "RFCXXXX: A YANG Data Model for Resource Reservation Protocol
    (RSVP)";
}

grouping graceful-restart-extended {
  description
    "Configuration parameters relating to RSVP Graceful-Restart.";
}

grouping authentication-extended {
  description
    "Configuration parameters relating to RSVP authentication.";
  leaf lifetime {
    type uint32 {
      range "30..86400";
    }
  }
}
```

```
    }
    units "seconds";
    default "30";
    description
      "Life time for each security association.";
    reference
      "RFC2747: RSVP Cryptographic Authentication";
  }
  leaf window-size {
    type uint32 {
      range "1..64";
    }
    default "2";
    description
      "Window-size to limit number of out-of-order messages.";
    reference
      "RFC2747: RSVP Cryptographic Authentication";
  }
  leaf challenge {
    type empty;
    description
      "Enable challenge messages.";
    reference
      "RFC2747: RSVP Cryptographic Authentication";
  }
  leaf retransmits {
    type uint32 {
      range "1..10000";
    }
    default "1";
    description
      "Number of retransmits when messages are dropped.";
    reference
      "RFC2747: RSVP Cryptographic Authentication";
  }
  leaf key-chain {
    type key-chain:key-chain-ref;
    description
      "Key chain name to authenticate RSVP
      signaling messages.";
    reference
      "RFC2747: RSVP Cryptographic Authentication";
  }
}

grouping hellos-extended {
  description
    "Configuration parameters relating to RSVP hellos";
```

```
leaf interface-based {
  type empty;
  description
    "Enable interface-based Hello adjacency if present.";
}
leaf hello-interval {
  type uint32;
  units "milliseconds";
  default "9000";
  description
    "Configure interval between successive Hello messages in
    milliseconds.";
  reference
    "RFC3209: RSVP-TE: Extensions to RSVP for LSP Tunnels.
    RFC5495: Description of the Resource Reservation Protocol -
    Traffic-Engineered (RSVP-TE) Graceful Restart Procedures.";
}
leaf hello-misses {
  type uint32 {
    range "1..10";
  }
  default "3";
  description
    "Configure max number of consecutive missed Hello messages.";
  reference
    "RFC3209: RSVP-TE: Extensions to RSVP for LSP Tunnels.
    RFC5495: Description of the Resource Reservation Protocol -
    Traffic- Engineered (RSVP-TE) Graceful Restart Procedures.";
}
}

grouping signaling-parameters-extended {
  description
    "Configuration parameters relating to RSVP signaling";
  leaf refresh-interval {
    type uint32;
    units "seconds";
    default "30";
    description
      "Set interval between successive refreshes";
    reference "RFC2205";
  }
  leaf refresh-misses {
    type uint32;
    default "9";
    description
      "Set max number of consecutive missed messages for state
      expiry";
  }
}
```

```
        reference "RFC2205";
    }
    leaf checksum-enable {
        type empty;
        description
            "Enable RSVP message checksum computation";
        reference "RFC2205";
    }
    leaf patherr-state-removal {
        type empty;
        description
            "State-Removal flag in Path Error message if present.";
        reference "RFC3473";
    }
}

grouping refresh-reduction-extended {
    description
        "Configuration parameters relating to RSVP refresh reduction.";
    leaf bundle-message-max-size {
        type uint32 {
            range "512..65000";
        }
        default "1500";
        description
            "Configure maximum size (bytes) of a single RSVP Bundle
            message.";
        reference "RFC2961";
    }
    leaf ack-hold-time {
        type uint32;
        units "milliseconds";
        default "9000";
        description
            "Configure hold time in milliseconds for sending RSVP ACK
            message(s).";
        reference "RFC2961";
    }
    leaf ack-max-size {
        type uint32;
        default "1500";
        description
            "Configure max size of a single RSVP ACK message.";
        reference "RFC2961";
    }
    leaf ack-retransmit-time {
        type uint32;
        units "milliseconds";
    }
}
```

```
    default "500";
    description
      "Configure min delay in milliseconds to wait for an
       acknowledgment before being retransmitted.";
    reference "RFC2961";
  }
  leaf srefresh-ack-desired {
    type empty;
    description
      "Enables the sending of MESSAGE_ID with ACK_Desired
       set with Srefresh messages.";
    reference "RFC2961";
  }
  leaf srefresh-max-size {
    type uint32 {
      range "20..65000";
    }
    default "1500";
    description
      "Configure max size (bytes) of a single RSVP Srefresh
       message.";
    reference "RFC2961";
  }
  leaf srefresh-relative-period {
    type uint8 {
      range "10..100";
    }
    description
      "Configures the period of Srefreshes relative to standard
       refresh message period in percentage.";
  }
}

grouping packets-extended-statistics {
  description
    "Packet statistics.";
  leaf discontinuity-time {
    type yang:date-and-time;
    description
      "The time on the most recent occasion at which any one or
       more of the statistic counters suffered a discontinuity.
       If no such discontinuities have occurred since the last
       re-initialization of the local management subsystem, then
       this node contains the time the local management subsystem
       re-initialized itself.";
  }
  leaf out-dropped {
    type yang:counter64;
  }
}
```

```
        description
            "Out RSVP packet drop count.";
    }
    leaf in-dropped {
        type yang:counter64;
        description
            "In RSVP packet drop count.";
    }
    leaf out-errors {
        type yang:counter64;
        description
            "Out RSVP packet errors count.";
    }
    leaf in-errors {
        type yang:counter64;
        description
            "In RSVP packet rx errors count.";
    }
}

/**
 * RSVP extensions augmentations
 */
/* RSVP graceful restart*/

augment "/rt:routing/rt:control-plane-protocols/"
    + "rt:control-plane-protocol/rsvp:rsvp/"
    + "rsvp:graceful-restart" {
    description
        "RSVP graceful restart configuration extensions";
    uses graceful-restart-extended;
}

/**
 * RSVP all interfaces extensions
 */

/* RSVP interface signaling extensions */
augment "/rt:routing/rt:control-plane-protocols/"
    + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces" {
    description
        "RSVP signaling all interfaces configuration extensions";
    uses signaling-parameters-extended;
}

/* Packet statistics extension */
augment "/rt:routing/rt:control-plane-protocols/"
```

```
    + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
    + "rsvp:statistics/rsvp:packets" {
  description
    "RSVP packets all interfaces configuration extensions";
  uses packets-extended-statistics;
}

/* RSVP refresh reduction extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:refresh-reduction" {
  description
    "RSVP refresh-reduction all interface configuration
    extensions";
  uses refresh-reduction-extended;
}

/* RSVP hellos extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:hellos" {
  description
    "RSVP hello all interfaces configuration extensions";
  uses hellos-extended;
}

/* RSVP authentication extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:authentication" {
  description
    "RSVP authentication all interfaces configuration extensions";
  uses authentication-extended;
}

/**
 * RSVP per interface extensions
 */
/* RSVP interface signaling extensions */

augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:interface" {
  description
    "RSVP signaling interface configuration extensions";
  uses signaling-parameters-extended;
}
```



```
/* Packet statistics extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:interface/rsvp:statistics/rsvp:packets" {
  description
    "RSVP packet stats extensions";
  uses packets-extended-statistics;
}

/* RSVP refresh reduction extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:interface/rsvp:refresh-reduction" {
  description
    "RSVP refresh-reduction interface configuration extensions";
  uses refresh-reduction-extended;
}

/* RSVP hellos extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:interface/rsvp:hellos" {
  description
    "RSVP hello interface configuration extensions";
  uses hellos-extended;
}

/* RSVP authentication extension */
augment "/rt:routing/rt:control-plane-protocols/"
  + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
  + "rsvp:interface/rsvp:authentication" {
  description
    "RSVP authentication interface configuration extensions";
  uses authentication-extended;
}
}
<CODE ENDS>
```

Figure 5: RSVP extended YANG module

## 6. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

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

URI: urn:ietf:params:xml:ns:yang:ietf-rsvp-extended  
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-rsvp  
namespace: urn:ietf:params:xml:ns:yang:ietf-rsvp  
prefix: rsvp  
reference: RFCXXXX

name: ietf-rsvp-extended  
namespace: urn:ietf:params:xml:ns:yang:ietf-rsvp-extended  
prefix: rsvp-extended  
reference: RFCXXXX

## 7. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in the YANG module(s) defined in this document 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:

/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/  
rsvp:rsvp/ /rsvp:globals /rsvp:interfaces /rsvp:sessions

All of which are considered sensitive and if access to either of these is compromised, it can result in temporary network outages or be employed to mount DoS attacks.

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:

```
/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/  
rsvp:rsvp/ /rsvp:globals /rsvp:interfaces /rsvp:sessions
```

Additional information from these state data nodes can be inferred with respect to the network topology, and device location and subsequently be used to mount other attacks in the network.

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

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

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

## 8. Acknowledgement

The authors would like to thank Tom Petch for reviewing and providing useful feedback about the document. The authors would also like to thank Lou Berger for reviewing and providing valuable feedback on this document.

## 9. Appendix A

A simple network setup is shown in {fig-example title}. R1 runs the RSVP routing protocol on both interfaces 'ge0/0/0/1', and 'ge0/0/0/2'.

State on R1:

Sessions:

=====

Destination	Protocol-ID	Dest-port
198.51.100.1	10	10

Neighbors:

=====

Neighbor Address	Interface
192.0.2.6	ge0/0/0/1

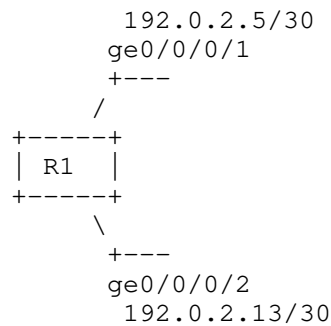


Figure 6: Example of network configuration.

The instance data tree could then be as follows:

```

{
  "ietf-routing:routing": {
    "control-plane-protocols": {
      "control-plane-protocol": [
        {
          "type": "rt:routing-protocol",
          "name": "rsvp:rsvp",
          "ietf-rsvp:rsvp": {
            "interfaces": {
              "refresh-reduction": {
                "enabled": true,
                "ietf-rsvp-extended:bundle-message-max-size": 2000,
                "ietf-rsvp-extended:reliable-ack-hold-time": 180,
                "ietf-rsvp-extended:reliable-ack-max-size": 2000,
                "ietf-rsvp-extended:reliable-retransmit-time": 180,
                "ietf-rsvp-extended:reliable-srefresh": [
                  null
                ],
                "ietf-rsvp-extended:summary-max-size": 2000
              }
            }
          }
        }
      ]
    }
  }
}

```

```
"hellos": {
  "enabled": true,
  "ietf-rsvp-extended:interface-based": [
    null
  ],
  "ietf-rsvp-extended:hello-interval": 27000,
  "ietf-rsvp-extended:hello-misses": 3
},
"statistics": {
  "messages": {
    "ack-sent": "777",
    "ack-received": "4840",
    "bundle-sent": "2195",
    "bundle-received": "293",
    "hello-sent": "2516",
    "hello-received": "3535",
    "integrity-challenge-sent": "2737",
    "integrity-challenge-received": "2330",
    "integrity-response-sent": "895",
    "integrity-response-received": "1029",
    "path-sent": "1197",
    "path-received": "3568",
    "path-err-sent": "4658",
    "path-err-received": "695",
    "path-tear-sent": "3706",
    "path-tear-received": "2604",
    "resv-sent": "3353",
    "resv-received": "3129",
    "resv-err-sent": "1787",
    "resv-err-received": "3205",
    "resv-tear-sent": "4465",
    "resv-tear-received": "3056",
    "summary-refresh-sent": "655",
    "summary-refresh-received": "3856"
  },
  "packets": {
    "sent": "2147",
    "received": "4374",
    "ietf-rsvp-extended:discontinuity-time":
      "2015-10-24T17:11:27+02:00",
    "ietf-rsvp-extended:out-dropped": "2696",
    "ietf-rsvp-extended:in-dropped": "941",
    "ietf-rsvp-extended:out-errors": "19",
    "ietf-rsvp-extended:in-errors": "2732"
  },
  "errors": {
    "authenticate": "2540",
    "checksum": "2566",
```

```
        "packet-length": "267"
      }
    },
    "interface": [
      {
        "interface": "ge0/0/0/1",
        "statistics": {
          "messages": {
            "ack-sent": "2747",
            "ack-received": "4934",
            "bundle-sent": "1618",
            "bundle-received": "3668",
            "hello-sent": "4288",
            "hello-received": "1194",
            "integrity-challenge-sent": "4850",
            "integrity-challenge-received": "3979",
            "integrity-response-sent": "479",
            "integrity-response-received": "1773",
            "path-sent": "2230",
            "path-received": "1793",
            "path-err-sent": "465",
            "path-err-received": "1859",
            "path-tear-sent": "923",
            "path-tear-received": "3924",
            "resv-sent": "3203",
            "resv-received": "2507",
            "resv-err-sent": "1259",
            "resv-err-received": "2445",
            "resv-tear-sent": "3045",
            "resv-tear-received": "4676",
            "summary-refresh-sent": "365",
            "summary-refresh-received": "2129"
          },
          "packets": {
            "sent": "847",
            "received": "3114",
            "ietf-rsvp-extended:discontinuity-time":
              "2015-10-24T17:11:27+02:00",
            "ietf-rsvp-extended:out-dropped": "1841",
            "ietf-rsvp-extended:in-dropped": "4832",
            "ietf-rsvp-extended:out-errors": "1334",
            "ietf-rsvp-extended:in-errors": "3900"
          },
          "errors": {
            "authenticate": "3494",
            "checksum": "4374",
            "packet-length": "2456"
          }
        }
      ]
    }
  }
}
```

```
    }
  },
  {
    "interface": "ge0/0/0/2",
    "statistics": {
      "messages": {
        "ack-sent": "1276",
        "ack-received": "2427",
        "bundle-sent": "4053",
        "bundle-received": "3509",
        "hello-sent": "3261",
        "hello-received": "2863",
        "integrity-challenge-sent": "4744",
        "integrity-challenge-received": "3554",
        "integrity-response-sent": "3155",
        "integrity-response-received": "169",
        "path-sent": "3853",
        "path-received": "409",
        "path-err-sent": "4227",
        "path-err-received": "2830",
        "path-tear-sent": "1742",
        "path-tear-received": "3344",
        "resv-sent": "3154",
        "resv-received": "3492",
        "resv-err-sent": "3112",
        "resv-err-received": "3974",
        "resv-tear-sent": "3657",
        "resv-tear-received": "533",
        "summary-refresh-sent": "4036",
        "summary-refresh-received": "2123"
      },
      "packets": {
        "sent": "473",
        "received": "314",
        "ietf-rsvp-extended:discontinuity-time":
          "2015-10-24T17:11:27+02:00",
        "ietf-rsvp-extended:out-dropped": "2042",
        "ietf-rsvp-extended:in-dropped": "90",
        "ietf-rsvp-extended:out-errors": "1210",
        "ietf-rsvp-extended:in-errors": "1361"
      },
      "errors": {
        "authenticate": "543",
        "checksum": "2241",
        "packet-length": "480"
      }
    }
  }
}
```

```
    ],
    "ietf-rsvp-extended:refresh-interval": 30,
    "ietf-rsvp-extended:refresh-misses": 5,
    "ietf-rsvp-extended:checksum_enabled": true,
    "ietf-rsvp-extended:patherr-state-removal": [
      null
    ]
  },
  "sessions": {
    "session-ip": [
      {
        "destination-port": 10,
        "protocol-id": 10,
        "destination": "198.51.100.1",
        "psbs": {
          "psb": [
            {
              "source-port": 10,
              "expires-in": 100
            }
          ]
        },
        "rsbs": {
          "rsb": [
            {
              "source-port": 10,
              "reservation-style":
                "rsvp:reservation-wildcard-filter",
              "expires-in": 100
            }
          ]
        }
      ]
    ]
  },
  "neighbors": {
    "neighbor": [
      {
        "address": "192.0.2.6",
        "epoch": 130,
        "expiry-time": 260,
        "graceful-restart": {
          "enabled": true,
          "local-restart-time": 271,
          "local-recovery-time": 138,
          "neighbor-restart-time": 341,
          "neighbor-recovery-time": 342
        }
      }
    ]
  }
}
```



```
        "hello-status": "enabled",
        "interface": "ge0/0/0/1",
        "restart-count": 2,
        "restart-time": "2015-10-24T17:11:27+02:00"
    }
}
],
},
},
},
}
```

Figure 7: Example RSVP JSON encoded data instance tree.

## 10. Contributors

Himanshu Shah  
Ciena

Email: hshah@ciena.com

Xia Chen  
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones  
Brocade

Email: raqib@Brocade.com

Bin Wen  
Comcast

Email: Bin\_Wen@cable.comcast.com

## 11. References

### 11.1. 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>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.

- [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>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [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>>.
- [RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <<https://www.rfc-editor.org/info/rfc8177>>.
- [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>>.
- [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>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.

- [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>>.

## 11.2. Informative References

- [RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <<https://www.rfc-editor.org/info/rfc2205>>.
- [RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, DOI 10.17487/RFC2747, January 2000, <<https://www.rfc-editor.org/info/rfc2747>>.
- [RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC 2961, DOI 10.17487/RFC2961, April 2001, <<https://www.rfc-editor.org/info/rfc2961>>.
- [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>>.
- [rfc4558] Ali, Z., Rahman, R., Prairie, D., and D. Papadimitriou, "Node-ID Based Resource Reservation Protocol (RSVP) Hello: A Clarification Statement", RFC 4558, DOI 10.17487/RFC4558, June 2006, <<https://www.rfc-editor.org/info/rfc4558>>.

- [RFC5063] Satyanarayana, A., Ed. and R. Rahman, Ed., "Extensions to GMPLS Resource Reservation Protocol (RSVP) Graceful Restart", RFC 5063, DOI 10.17487/RFC5063, October 2007, <<https://www.rfc-editor.org/info/rfc5063>>.
- [RFC5495] Li, D., Gao, J., Satyanarayana, A., and S. Bardalai, "Description of the Resource Reservation Protocol - Traffic-Engineered (RSVP-TE) Graceful Restart Procedures", RFC 5495, DOI 10.17487/RFC5495, March 2009, <<https://www.rfc-editor.org/info/rfc5495>>.

## Authors' Addresses

Vishnu Pavan Beeram  
Juniper Networks

Email: [vbeeram@juniper.net](mailto:vbeeram@juniper.net)

Tarek Saad  
Juniper Networks

Email: [tsaad@juniper.net](mailto:tsaad@juniper.net)

Rakesh Gandhi  
Cisco Systems, Inc.

Email: [rgandhi@cisco.com](mailto:rgandhi@cisco.com)

Xufeng Liu  
Volta Networks

Email: [xufeng.liu.ietf@gmail.com](mailto:xufeng.liu.ietf@gmail.com)

Igor Bryskin  
Individual

Email: [i\\_bryskin@yahoo.com](mailto:i_bryskin@yahoo.com)

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.

## 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].

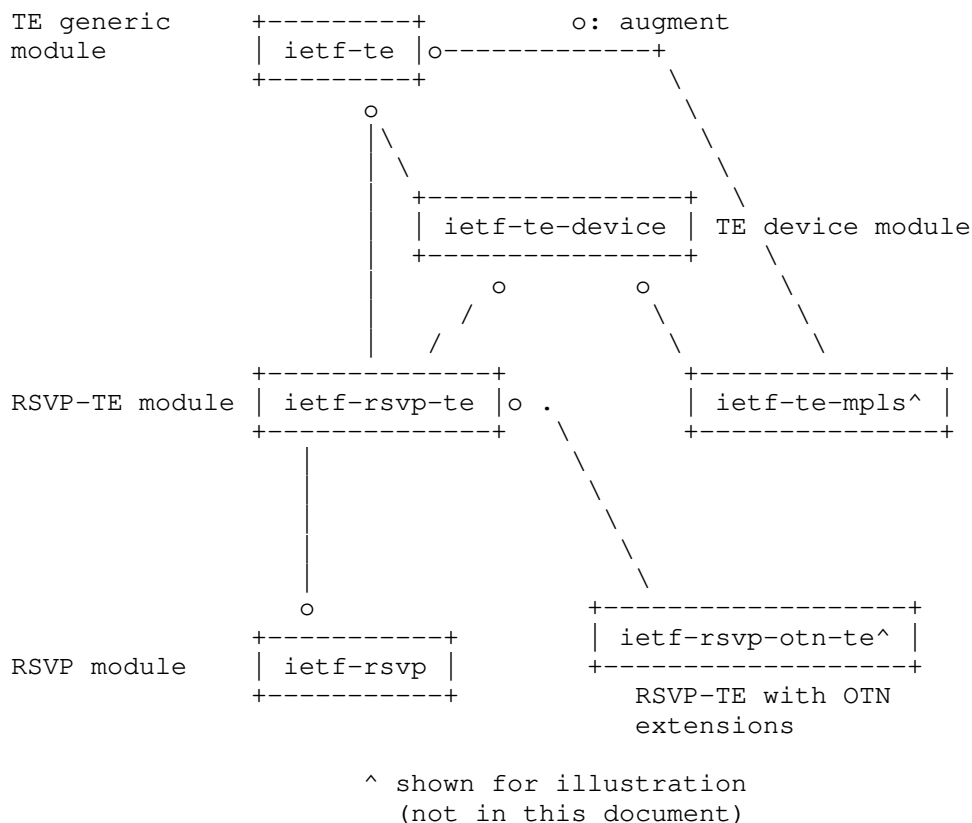


Figure 1: Relationship of TE module(s) with signaling protocol modules

## 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.

**globals:**

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?
|               |
|               +--rw 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 leafs 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 leafs that define the tunnel source and destination node endpoints.

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

YANG leafs 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.

```

|      +--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 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 (technology)?
            +--:(generic)
              +--rw generic? te-bandwidth
        +--rw link-protection? identityref
        +--rw setup-priority? uint8
        +--rw hold-priority? uint8
        +--rw signaling-type? identityref
        +--rw path-metric-bounds
          +--rw path-metric-bound* [metric-type]
            +--rw metric-type identityref
            +--rw upper-bound? uint64
        +--rw path-affinities-values
          +--rw path-affinities-value* [usage]
            +--rw usage identityref
            +--rw value? admin-groups
        +--rw path-affinity-names
          +--rw path-affinity-name* [usage]
            +--rw usage identityref
            +--rw affinity-name* [name]
              +--rw name string
        +--rw path-srlgs-lists
          +--rw path-srlgs-list* [usage]
            +--rw usage identityref
            +--rw values* srlg
        +--rw path-srlgs-names
          +--rw path-srlgs-name* [usage]
            +--rw usage identityref
            +--rw names* string
        +--rw disjointness?

```

```

    te-path-disjointness
+--rw explicit-route-objects-always
+--rw route-object-exclude-always* [index]
+--rw index                               uint32
+--rw (type)?
+--:(numbered-node-hop)
+--rw numbered-node-hop
+--rw 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?
+--rw rt-types:generalized-label
+--rw direction?
+--rw 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

```

```

+--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
    +--rw 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 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)
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--rw generic?
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   rt-types:ge
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   +--rw direction?
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   te-label-directio
|   |   +--:(srlg)
|   |   |
|   |   |   +--rw srlg
|   |   |   |
|   |   |   |   +--rw srlg?   uint32

```

```

+---rw explicit-route-include-objects
+---rw route-object-include-object*
    [index]
+---rw index
    |   uint32
+---rw (type)?
+---:(numbered-node-hop)
    +---rw numbered-node-hop
        +---rw node-id
            |   te-node-id
        +---rw hop-type?
            |   te-hop-type
+---:(numbered-link-hop)
    +---rw numbered-link-hop
        +---rw link-tp-id
            |   te-tp-id
        +---rw hop-type?
            |   te-hop-type
        +---rw direction?
            |   te-link-direction
+---:(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:ge
neralized-label
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 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

```

bel

```

+---:(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
+---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

```

bel

```

+---:(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
|           +---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
+--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
+--rw primary-reverse-path
  +--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 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)
|                               +---rw generic?
|                                   rt-types
:generalized-label
+---rw direction?
    te-label-direc

```

tion

```

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

```



```

tion
}

    +---rw direction?
        te-label-direct

    +---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 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?
+---rw                      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?
+---rw                      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?
+---rw                      rt-types:generalized
- label
+---rw direction?
+---rw                      te-label-direction
+---rw route-object-include-exclude* [index]
+---rw explicit-route-usage?
+---rw 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> +---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				<pre> +---rw direction? </pre>

					<pre> 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 </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					
					<pre> +--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] </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
                    +---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:ge

neralized-label

+---ro direction?
te-label-directio

n

+---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
+---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 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)
|       |       +---rw generic?
|       rt-types:ge
+---rw direction?
|       te-label-direction

+---:(srlg)
+---rw srlg
|       +---rw srlg?    uint32
+---rw explicit-route-include-objects
+---rw route-object-include-object*
|       [index]
+---rw index
|       uint32
+---rw (type)?
+---:(numbered-node-hop)
+---rw numbered-node-hop
+---rw node-id
|       te-node-id
+---rw hop-type?

```



```

| | | | | | | te-hop-type
| | | | | | | +---:(numbered-link-hop)
| | | | | | | | +--rw numbered-link-hop
| | | | | | | | | +--rw link-tp-id
| | | | | | | | | | te-tp-id
| | | | | | | | | +--rw hop-type?
| | | | | | | | | | te-hop-type
| | | | | | | | | +--rw direction?
| | | | | | | | | | te-link-direction
| | | | | | | +---:(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: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 values*      srlg
+---rw path-srlgs-names
+---rw path-srlgs-name* [usage]
+---rw usage      identityref
+---rw names*      string
+---rw disjointness?
+---rw te-path-disjointness
+---rw explicit-route-objects-always
+---rw route-object-exclude-always* [index]
+---rw index      uint32
+---rw (type)?
+---:(numbered-node-hop)
+---rw numbered-node-hop
+---rw node-id      te-node-id
+---rw hop-type?      te-hop-type
+---:(numbered-link-hop)
+---rw numbered-link-hop
+---rw link-tp-id      te-tp-id
+---rw hop-type?      te-hop-type
+---rw direction?      te-link-direction
+---:(unnumbered-link-hop)
+---rw unnumbered-link-hop
+---rw link-tp-id      te-tp-id
+---rw node-id      te-node-id
+---rw hop-type?      te-hop-type
+---rw direction?      te-link-direction

```

				<pre> +--:(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 </pre>
bel				<pre> +--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-la </pre>
bel				<pre> +--rw direction?     te-label-direction +--:(srlg) </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
            +---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 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
				+--ro (technology)?
				+--:(generic)
				+--ro generic?   te-bandwidth
				+--ro disjointness-type?
				te-types:te-path-disjointness
				+--ro computed-path-error-infos

alized-label

```

    +--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
+--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)

```



```

+---rw generic?
      rt-types:generic
n
+---rw direction?
      te-label-direction
+---:(srlg)
  +---rw srlg
  +---rw srlg?   uint32
+---rw explicit-route-include-objects
  +---rw route-object-include-object*
    [index]
  +---rw index
    |   uint32
  +---rw (type)?
    +---:(numbered-node-hop)
      +---rw numbered-node-hop
      +---rw node-id
      |   te-node-id
      +---rw hop-type?
      |   te-hop-type
    +---:(numbered-link-hop)
      +---rw numbered-link-hop
      +---rw link-tp-id
      |   te-tp-id
      +---rw hop-type?
      |   te-hop-type
      +---rw direction?
      |   te-link-direction
    +---:(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:generic
n
+---rw direction?
te-label-direction
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 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
			+--:	(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)

bel

```

    +--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
            +--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 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 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/lsp/lsp/tunnel-name
|       +---ro node              -> /te/lsp/lsp/node
|       +---ro lsp-id             -> /te/lsp/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 lsp
+---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)

```



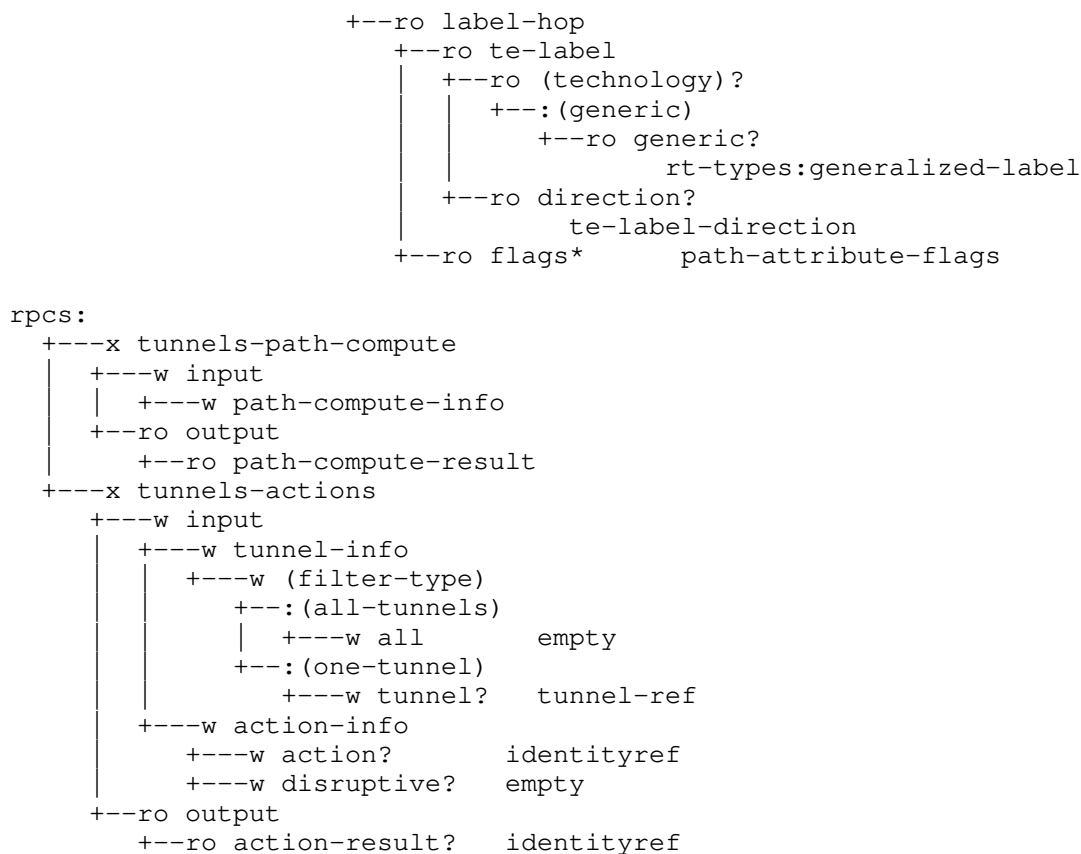


Figure 8: TE Tunnel generic model YANG tree diagram

### 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].

```
<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>
```

```
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 lsp-grouping {
      description
        "LSPs state operational data grouping.";
      container lsp {
        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.";
      }
    }
  }
}
```

```
    }
  }
  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 lsp-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/lspss"`: 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.

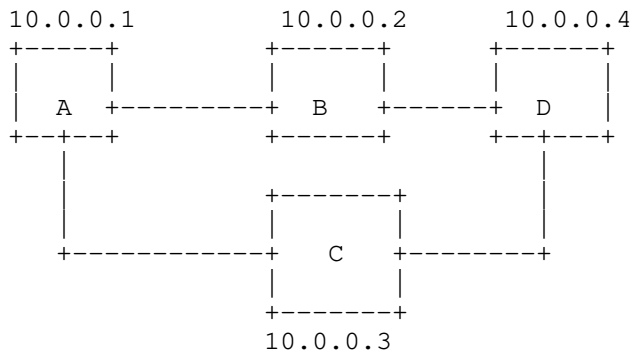


Figure 13: TE network used in data tree examples

### 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"
    }
  ]
}

```

### 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, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.

- [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>>.

- [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

- [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, 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, DOI 10.17487/RFC9012, April 2021, <<https://www.rfc-editor.org/info/rfc9012>>.

#### Authors' Addresses

Tarek Saad  
Juniper Networks  
  
Email: [tsaad@juniper.net](mailto:tsaad@juniper.net)

Rakesh Gandhi  
Cisco Systems Inc  
  
Email: [rgandhi@cisco.com](mailto:rgandhi@cisco.com)

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

TEAS Working Group  
Internet Draft  
Intended status: Standards Track

Xufeng Liu  
Volta Networks  
Igor Bryskin  
Huawei Technologies  
Vishnu Pavan Beeram  
Tarek Saad  
Juniper Networks  
Himanshu Shah  
Ciena  
Oscar Gonzalez De Dios  
Telefonica

Expires: December 19, 2019

June 19, 2019

YANG Data Model for Traffic Engineering (TE) Topologies  
draft-ietf-teas-yang-te-topo-22

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

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

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

This Internet-Draft will expire on December 19, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Abstract

This document defines a YANG data model for representing, retrieving and manipulating Traffic Engineering (TE) Topologies. The model serves as a base model that other technology specific TE Topology models can augment.

## Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## Table of Contents

|                                                           |    |
|-----------------------------------------------------------|----|
| 1. Introduction.....                                      | 3  |
| 1.1. Terminology.....                                     | 4  |
| 1.2. Tree Structure.....                                  | 4  |
| 1.3. Prefixes in Data Node Names.....                     | 5  |
| 2. Characterizing TE Topologies.....                      | 5  |
| 3. Modeling Abstractions and Transformations.....         | 7  |
| 3.1. TE Topology.....                                     | 7  |
| 3.2. TE Node.....                                         | 7  |
| 3.3. TE Link.....                                         | 8  |
| 3.4. Transitional TE Link for Multi-Layer Topologies..... | 8  |
| 3.5. TE Link Termination Point (LTP).....                 | 10 |
| 3.6. TE Tunnel Termination Point (TTP).....               | 10 |
| 3.7. TE Node Connectivity Matrix.....                     | 11 |
| 3.8. TTP Local Link Connectivity List (LLCL).....         | 11 |
| 3.9. TE Path.....                                         | 11 |
| 3.10. TE Inter-Layer Lock.....                            | 12 |
| 3.11. Underlay TE topology.....                           | 13 |
| 3.12. Overlay TE topology.....                            | 13 |
| 3.13. Abstract TE topology.....                           | 13 |
| 4. Model Applicability.....                               | 14 |
| 4.1. Native TE Topologies.....                            | 14 |



|                                                                                      |     |
|--------------------------------------------------------------------------------------|-----|
| 4.2. Customized TE Topologies.....                                                   | 16  |
| 4.3. Merging TE Topologies Provided by Multiple Providers.....                       | 19  |
| 4.4. Dealing with Multiple Abstract TE Topologies Provided by the Same Provider..... | 22  |
| 5. Modeling Considerations.....                                                      | 25  |
| 5.1. Network topology building blocks.....                                           | 25  |
| 5.2. Technology agnostic TE Topology model.....                                      | 25  |
| 5.3. Model Structure.....                                                            | 26  |
| 5.4. Topology Identifiers.....                                                       | 27  |
| 5.5. Generic TE Link Attributes.....                                                 | 27  |
| 5.6. Generic TE Node Attributes.....                                                 | 28  |
| 5.7. TED Information Sources.....                                                    | 29  |
| 5.8. Overlay/Underlay Relationship.....                                              | 30  |
| 5.9. Templates.....                                                                  | 31  |
| 5.10. Scheduling Parameters.....                                                     | 32  |
| 5.11. Notifications.....                                                             | 33  |
| 6. Guidance for Writing Technology Specific TE Topology Augmentations .....          | 33  |
| 7. TE Topology YANG Module.....                                                      | 46  |
| 8. Security Considerations.....                                                      | 92  |
| 9. IANA Considerations.....                                                          | 94  |
| 10. References.....                                                                  | 95  |
| 10.1. Normative References.....                                                      | 95  |
| 10.2. Informative References.....                                                    | 96  |
| 11. Acknowledgments.....                                                             | 100 |
| Appendix A. Complete Model Tree Structure.....                                       | 101 |
| Appendix B. Companion YANG Model for Non-NMDA Compliant Implementations.....         | 163 |
| Appendix C. Example: YANG Model for Technology Specific Augmentations .....          | 172 |
| Contributors.....                                                                    | 210 |
| Authors' Addresses.....                                                              | 210 |

## 1. Introduction

The Traffic Engineering Database (TED) is an essential component of Traffic Engineered (TE) systems that are based on MPLS-TE [RFC2702] and GMPLS [RFC3945]. The TED is a collection of all TE information about all TE nodes and TE links in the network. The TE Topology is a schematic arrangement of TE nodes and TE links present in a given TED. There could be one or more TE Topologies present in a given Traffic Engineered system. A TE Topology is the topology on which path computational algorithms are run to compute Traffic Engineered Paths (TE Paths).

This document defines a YANG [RFC7950] data model for representing and manipulating TE Topologies. This model contains technology

agnostic TE Topology building blocks that can be augmented and used by other technology-specific TE Topology models.

### 1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The reader is assumed to be familiar with general body of work captured in currently available TE related RFCs. [RFC7926] serves as a good starting point for those who may be less familiar with Traffic Engineering related RFCs.

Some of the key terms used in this document are:

**TED:** The Traffic Engineering Database is a collection of all TE information about all TE nodes and TE links in a given network.

**TE-Topology:** The TE Topology is a schematic arrangement of TE nodes and TE links in a given TED. It forms the basis for a graph suitable for TE path computations.

**Native TE Topology:** Native TE Topology is a topology that is native to a given provider network. Native TE topology could be discovered via various routing protocols and/or subscribe/publish techniques. This is the topology on which path computational algorithms are run to compute TE Paths.

**Customized TE Topology:** Customized TE Topology is a custom topology that is produced by a provider for a given client. This topology typically makes abstractions on the provider's Native TE Topology, and is provided to the client. The client receives the Customized TE Topology, and merges it into the client's Native TE Topology. The client's path computational algorithms aren't typically run on the Customized TE Topology; they are run on the client's Native TE Topology after the merge.

### 1.2. Tree Structure

A simplified graphical representation of the data model is presented in Appendix A. of this document. The tree format defined in [RFC8340] is used for the YANG data model tree representation.

### 1.3. Prefixes in Data Node Names

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

| Prefix   | YANG module           | Reference                     |
|----------|-----------------------|-------------------------------|
| yang     | ietf-yang-types       | [RFC6991]                     |
| inet     | ietf-inet-types       | [RFC6991]                     |
| nw       | ietf-network          | [RFC6991]                     |
| nt       | ietf-network-topology | [RFC8345]                     |
| te-types | ietf-te-types         | [I-D.ietf-teas-yang-te-types] |

Table 1: Prefixes and corresponding YANG modules

## 2. Characterizing TE Topologies

The data model proposed by this document takes the following characteristics of TE Topologies into account:

- TE Topology is an abstract control-plane representation of the data-plane topology. Hence attributes specific to the data-plane must make their way into the corresponding TE Topology modeling. The TE Topology comprises of dynamic auto-discovered data as well as fairly static data associated with data-plane nodes and links. The dynamic data may change frequently, such as unreserved bandwidth available on data-plane links. The static data rarely changes, such as layer network identification, switching and adaptation capabilities and limitations, fate sharing, and administrative colors. It is possible for a single TE Topology to encompass TE information at multiple switching layers.
- TE Topologies are protocol independent. Information about topological elements may be learnt via link-state protocols, but the topology can exist without being dependent on any particular protocol.
- TE Topology may not be congruent to the routing topology in a given TE System. The routing topology is constructed based on routing adjacencies. There isn't always a one-to-one association between a TE-link and a routing adjacency. For example, the presence of a TE link between a pair of nodes doesn't necessarily imply the existence of a routing-adjacency between these nodes. To

learn more, see [I-D.ietf-teas-te-topo-and-tunnel-modeling] and [I-D.ietf-teas-yang-l3-te-topo].

- Each TE Topological element has at least one information source associated with it. In some scenarios, there could be more than one information source associated with any given topological element.
- TE Topologies can be hierarchical. Each node and link of a given TE Topology can be associated with respective underlay topology. This means that each node and link of a given TE Topology can be associated with an independent stack of supporting TE Topologies.
- TE Topologies can be customized. TE topologies of a given network presented by the network provider to its client could be customized on per-client request basis. This customization could be performed by provider, by client or by provider/client negotiation. The relationship between a customized topology and provider's native topology could be captured as hierarchical (overlay-underlay), but otherwise the two topologies are decoupled from each other. A customized topology is presented to the client, while provider's native topology is known in its entirety to the provider itself.

### 3. Modeling Abstractions and Transformations

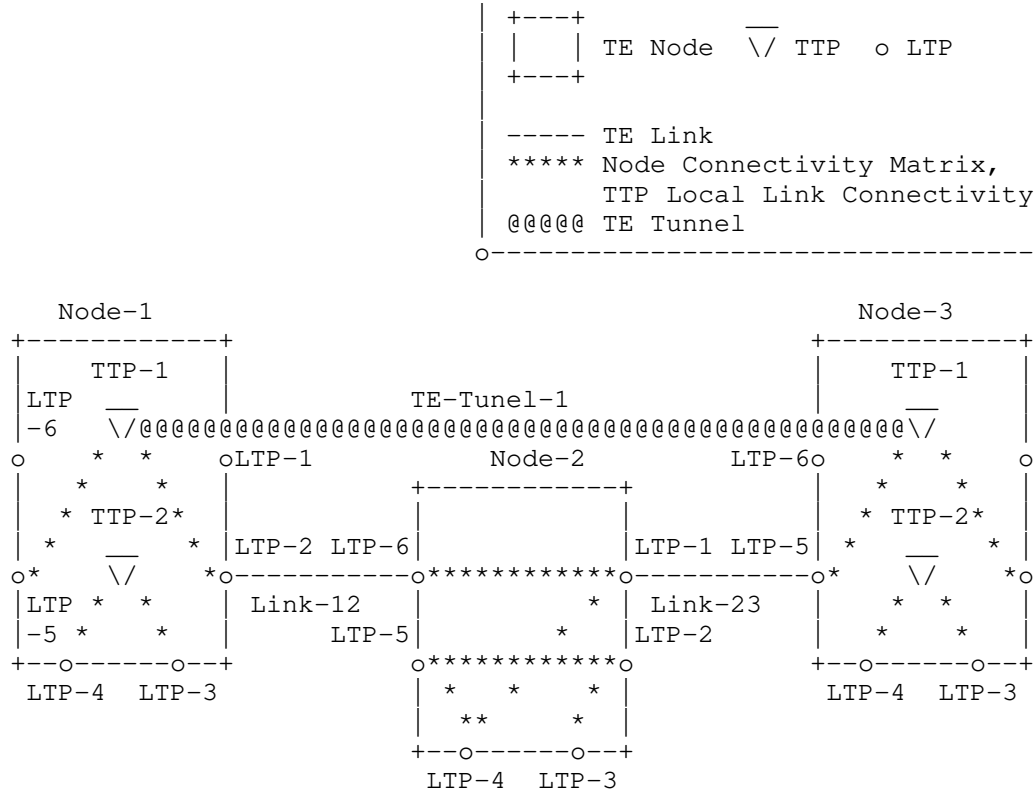


Figure 1: TE Topology Modeling Abstractions

#### 3.1. TE Topology

TE topology is a traffic engineering representation of one or more layers of network topologies. TE topology is comprised of TE nodes (TE graph vertices) interconnected via TE links (TE graph edges). A TE topology is mapped to a TE graph.

#### 3.2. TE Node

TE node is an element of a TE topology, presented as a vertex on TE graph. TE node represents one or several nodes, or a fraction of a node, which can be a switch or router that is physical or virtual. TE node belongs to and is fully defined in exactly one TE topology. TE node is assigned a unique ID within the TE topology scope. TE node attributes include information related to the data plane aspects of

the associated node(s) (e.g. connectivity matrix), as well as configuration data (such as TE node name). A given TE node can be reached on the TE graph over one of TE links terminated by the TE node.

Multi-layer TE nodes providing switching functions at multiple network layers are an example where a physical node can be decomposed into multiple logical TE nodes, which are fractions of the physical node. Some of these (logical) TE nodes may reside in the client layer TE topology while the remaining TE nodes belong to the server layer TE topology.

In Figure 1, Node-1, Node-2, and Node-3 are TE nodes.

### 3.3. TE Link

TE link is an element of a TE topology, presented as an edge on TE graph. The arrows on an edge indicate one or both directions of the TE link. When there are a pair of parallel links of opposite directions, an edge without arrows is also used. TE link represents one or several (physical) links or a fraction of a link. TE link belongs to and is fully defined in exactly one TE topology. TE link is assigned a unique ID within the TE topology scope. TE link attributes include parameters related to the data plane aspects of the associated link(s) (e.g. unreserved bandwidth, resource maps/pools, etc.), as well as the configuration data (such as remote node/link IDs, SRLGs, administrative colors, etc.). TE link is connected to TE node, terminating the TE link via exactly one TE link termination point (LTP).

In Figure 1, Link-12 and Link-23 are TE links.

### 3.4. Transitional TE Link for Multi-Layer Topologies

Networks are typically composed of multiple network layers where one or multiple signals in the client layer network can be multiplexed and encapsulated into a server layer signal [RFC5212] [G.805]. The server layer signal can be carried in the server layer network across multiple nodes until the server layer signal is terminated and the client layer signals reappear in the node that terminates the server layer signal. Examples of multi-layer networks are: IP over MPLS over Ethernet, low order Optical Data Unit-k (ODUk) signals multiplexed into a high order ODUL (l>k) carried over an Optical Channel (OCh) signal in an optical transport network as defined in [G.872] and [G.709].

TE links as defined in Section 3.3. can be used to represent links within a network layer. In case of a multi-layer network, TE nodes and TE links only allow representation of each network layer as a separate TE topology. Each of these single layer TE topologies would be isolated from their client and their server layer TE topology, if present. The highest and the lowest network layer in the hierarchy only have a single adjacent layer below or above, respectively. Multiplexing of client layer signals and encapsulating them into a server layer signal requires a function that is provided inside a node (typically realized in hardware). This function is also called layer transition.

One of the key requirements for path computation is to be able to calculate a path between two endpoints across a multi-layer network based on the TE topology representing this multi-layer network. This means that an additional TE construct is needed that represents potential layer transitions in the multi-layer TE-topology that connects the TE-topologies representing each separate network layer. The so-called transitional TE link is such a construct and it represents the layer transition function residing inside a node that is decomposed into multiple logical nodes that are represented as TE nodes (see also the transitional link definition in [G.8080] for the optical transport network). Hence, a transitional TE link connects a client layer node with a server layer node. A TE link as defined in 3.3. has LTPs of exactly the same kind on each link end whereas the transitional TE link has client layer LTPs on the client side of the transitional link and in most cases a single server layer LTP on the server side. It should be noted that transitional links are a helper construct in the multi-layer TE topology and they only exist as long as they are not in use, as they represent potential connectivity. When the server layer trail has been established between the server layer LTP of two transitional links in the server layer network, the resulting client layer link in the data plane will be represented as a normal TE link in the client layer topology. The transitional TE links will re-appear when the server layer trail has been torn down.

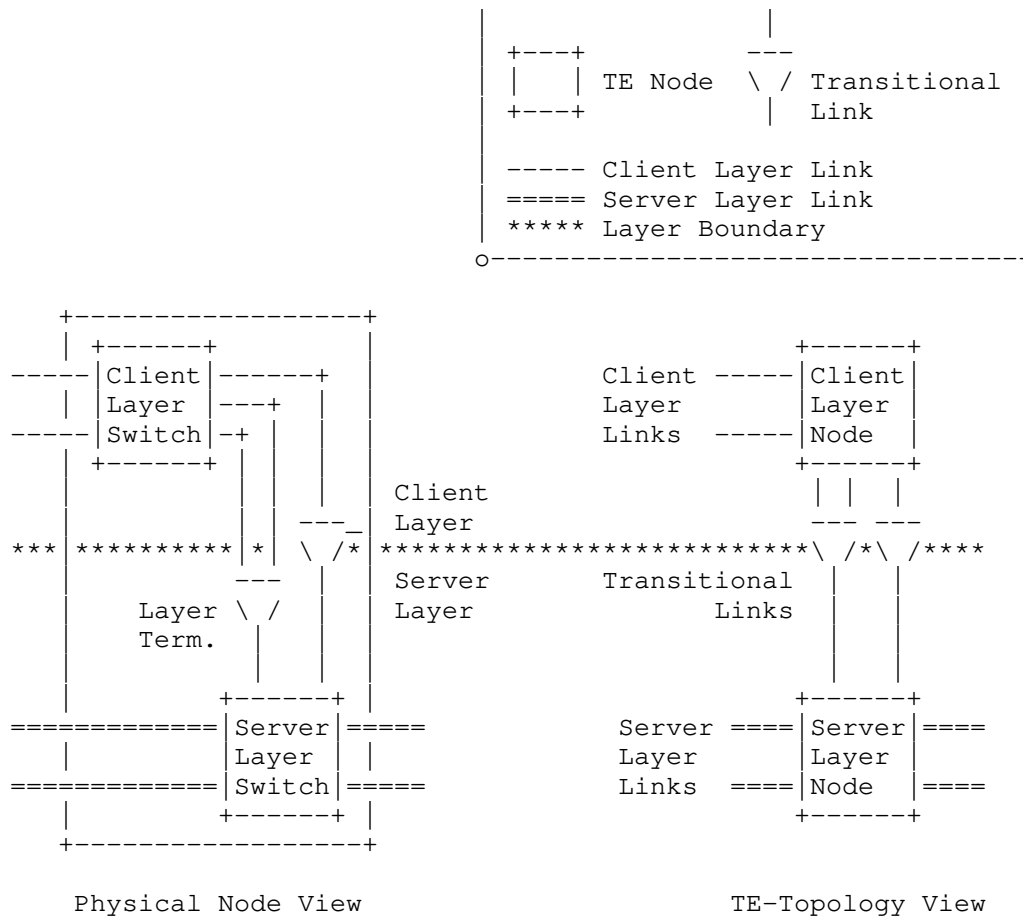


Figure 2: Modeling a Multi-Layer Node (Dual-Layer Example)

### 3.5. TE Link Termination Point (LTP)

TE link termination point (LTP) is a conceptual point of connection of a TE node to one of the TE links, terminated by the TE node. Cardinality between an LTP and the associated TE link is 1:0..1.

In Figure 1, Node-2 has six LTPs: LTP-1 to LTP-6.

### 3.6. TE Tunnel Termination Point (TTP)

TE tunnel termination point (TTP) is an element of TE topology representing one or several of potential transport service termination points (i.e. service client adaptation points such as



WDM/OCh transponder). TTP is associated with (hosted by) exactly one TE node. TTP is assigned a unique ID within the TE node scope. Depending on the TE node's internal constraints, a given TTP hosted by the TE node could be accessed via one, several or all TE links terminated by the TE node.

In Figure 1, Node-1 has two TTPs: TTP-1 and TTP-2.

### 3.7. TE Node Connectivity Matrix

TE node connectivity matrix is a TE node's attribute describing the TE node's switching limitations in a form of valid switching combinations of the TE node's LTPs (see below). From the point of view of a potential TE path arriving at the TE node at a given inbound LTP, the node's connectivity matrix describes valid (permissible) outbound LTPs for the TE path to leave the TE node from.

In Figure 1, the connectivity matrix on Node-2 is:  
{<LTP-6, LTP-1>, <LTP-5, LTP-2>, <LTP-5, LTP-4>, <LTP-4, LTP-1>, <LTP-3, LTP-2>}

### 3.8. TTP Local Link Connectivity List (LLCL)

TTP Local Link Connectivity List (LLCL) is a List of TE links terminated by the TTP hosting TE node (i.e. list of the TE link LTPs), which the TTP could be connected to. From the point of view of a potential TE path, LLCL provides a list of valid TE links the TE path needs to start/stop on for the connection, taking the TE path, to be successfully terminated on the TTP in question.

In Figure 1, the LLCL on Node-1 is:  
{<TTP-1, LTP-5>, <TTP-1, LTP-2>, <TTP-2, LTP-3>, <TTP-2, LTP4>}

### 3.9. TE Path

TE path is an ordered list of TE links and/or TE nodes on the TE topology graph, inter-connecting a pair of TTPs to be taken by a potential connection. TE paths, for example, could be a product of successful path computation performed for a given transport service.

In Figure 1, the TE Path for TE-Tunnel-1 is:  
{Node-1:TTP-1, Link-12, Node-2, Link-23, Node-3:TTP1}

## 3.10. TE Inter-Layer Lock

TE inter-layer lock is a modeling concept describing client-server layer adaptation relationships and hence important for the multi-layer traffic engineering. It is an association of M client layer LTPs and N server layer TTPs, within which data arriving at any of the client layer LTPs could be adopted onto any of the server layer TTPs. TE inter-layer lock is identified by inter-layer lock ID, which is unique across all TE topologies provided by the same provider. The client layer LTPs and the server layer TTPs associated within a given TE inter-layer lock are annotated with the same inter-layer lock ID attribute.

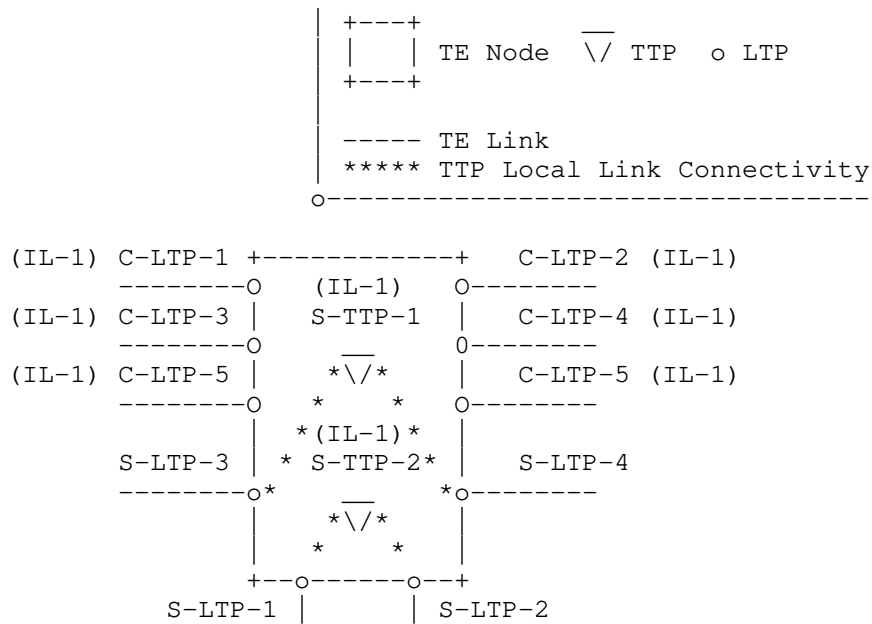


Figure 3: TE Inter-Layer Lock ID Associations

On the picture above a TE inter-layer lock with IL\_1 ID associates 6 client layer LTPs (C-LTP-1 - C-LTP-6) with two server layer TTPs (S-TTP-1 and S-TTP-2). They all have the same attribute - TE inter-layer lock ID: IL-1, which is the only thing that indicates the association. A given LTP may have 0, 1 or more inter-layer lock IDs. In the latter case this means that the data arriving at the LTP may be adopted onto any of TTPs associated with all specified inter-layer locks. For example, C-LTP-1 could have two inter-layer lock IDs - IL-1 and IL-2. This would mean that C-LTP-1 for adaptation purposes could use not just TTPs associated with inter-layer lock IL-1 (i.e.

S-TTP-1 and S-TTP-2 on the picture), but any of TTPs associated with inter-layer lock IL-2 as well. Likewise, a given TTP may have one or more inter-layer lock IDs, meaning that it can offer the adaptation service to any of client layer LTPs with inter-layer lock ID matching one of its own. Additionally, each TTP has an attribute - Unreserved Adaptation Bandwidth, which announces its remaining adaptation resources sharable between all potential client LTPs.

LTPs and TTPs associated within the same TE inter-layer lock may be hosted by the same (hybrid, multi-layer) TE node or multiple TE nodes located in the same or separate TE topologies. The latter is especially important since TE topologies of different layer networks could be modeled by separate augmentations of the basic (common to all layers) TE topology model.

### 3.11. Underlay TE topology

Underlay TE topology is a TE topology that serves as a base for constructing of overlay TE topologies

### 3.12. Overlay TE topology

Overlay TE topology is a TE topology constructed based on one or more underlay TE topologies. Each TE node of the overlay TE topology represents an arbitrary segment of an underlay TE topology; each TE link of the overlay TE topology represents an arbitrary TE path in one of the underlay TE topologies. The overlay TE topology and the supporting underlay TE topologies may represent distinct layer networks (e.g. OTN/ODUk and WDM/OCh respectively) or the same layer network.

### 3.13. Abstract TE topology

Abstract TE topology is a topology that contains abstract topological elements (nodes, links, tunnel termination points). Abstract TE topology is an overlay TE topology created by a topology provider and customized for a topology provider's client based on one or more of the provider's native TE topologies (underlay TE topologies), the provider's policies and the client's preferences. For example, a first level topology provider (such as Domain Controller) can create an abstract TE topology for its client (e.g. Multi-Domain Service Coordinator) based on the provider's one or more native TE topologies, local policies/profiles and the client's TE topology configuration requests

Figure 4 shows an example of abstract TE topology.

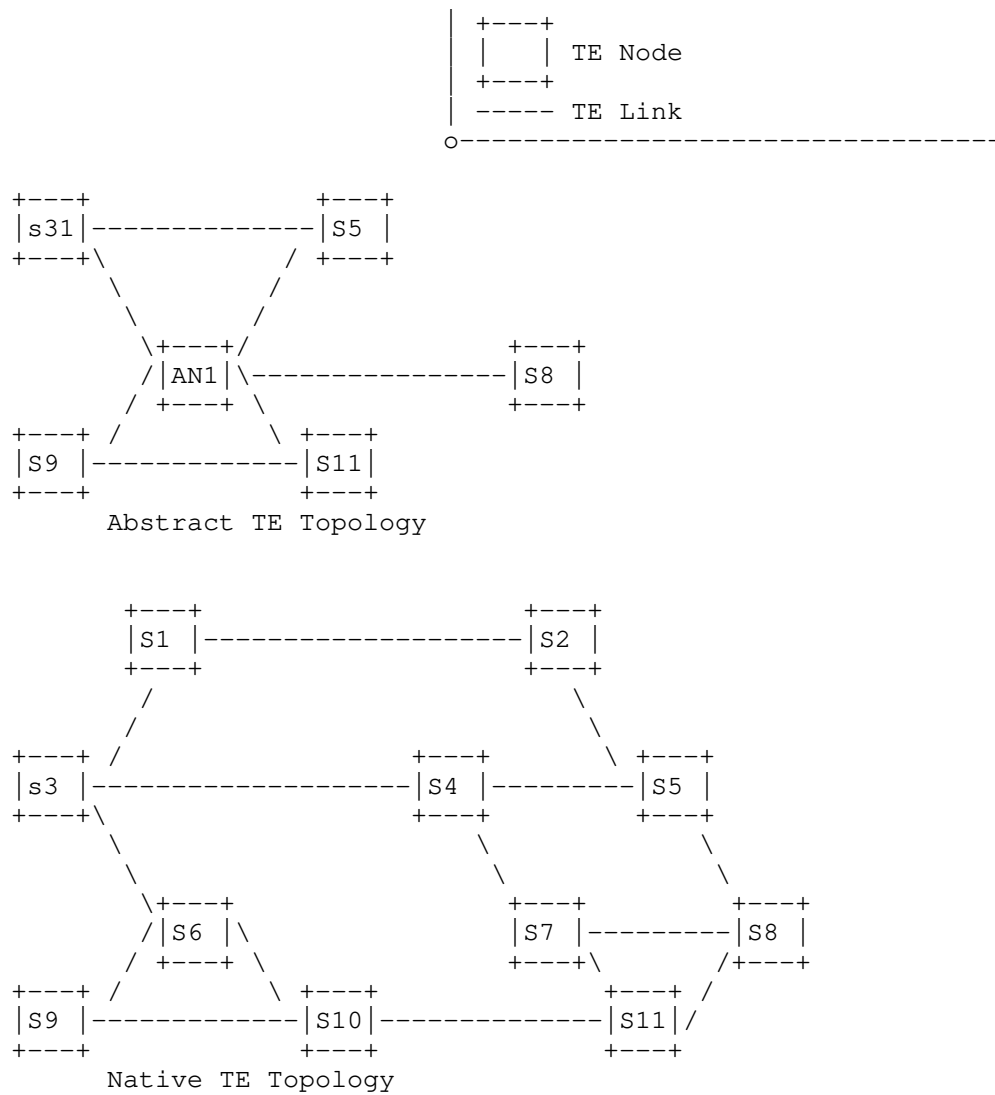


Figure 4: Abstract TE Topology

#### 4. Model Applicability

##### 4.1. Native TE Topologies

The model discussed in this draft can be used to represent and retrieve native TE topologies on a given TE system.

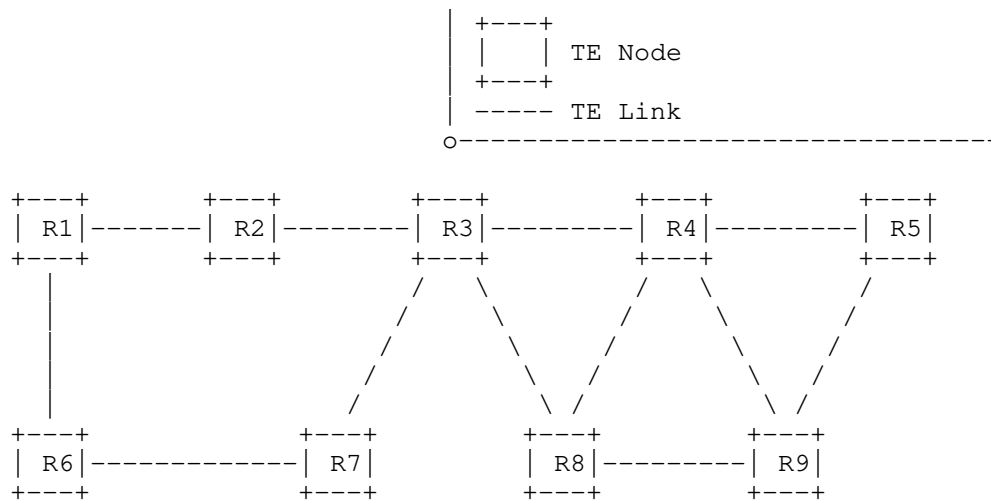


Figure 5a: Example Network Topology

Consider the network topology depicted in Figure 5a. R1 .. R9 are nodes representing routers. An implementation MAY choose to construct a native TE Topology using all nodes and links present in the given TED as depicted in Figure 5b. The data model proposed in this document can be used to retrieve/represent this TE topology.

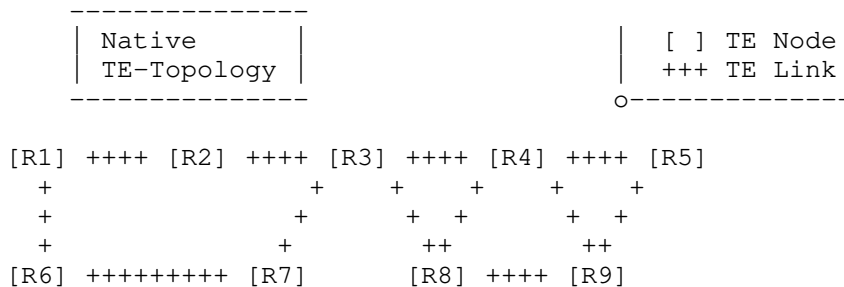


Figure 5b: Native TE Topology as seen on Node R3

Consider the case of the topology being split in a way that some nodes participate in OSPF-TE while others participate in ISIS-TE (Figure 6a). An implementation MAY choose to construct separate TE Topologies based on the information source. The native TE Topologies constructed using only nodes and links that were learnt via a specific information source are depicted in Figure 6b. The data model proposed in this document can be used to retrieve/represent these TE topologies.

Similarly, the data model can be used to represent/retrieve a TE Topology that is constructed using only nodes and links that belong to a particular technology layer. The data model is flexible enough to retrieve and represent many such native TE Topologies.

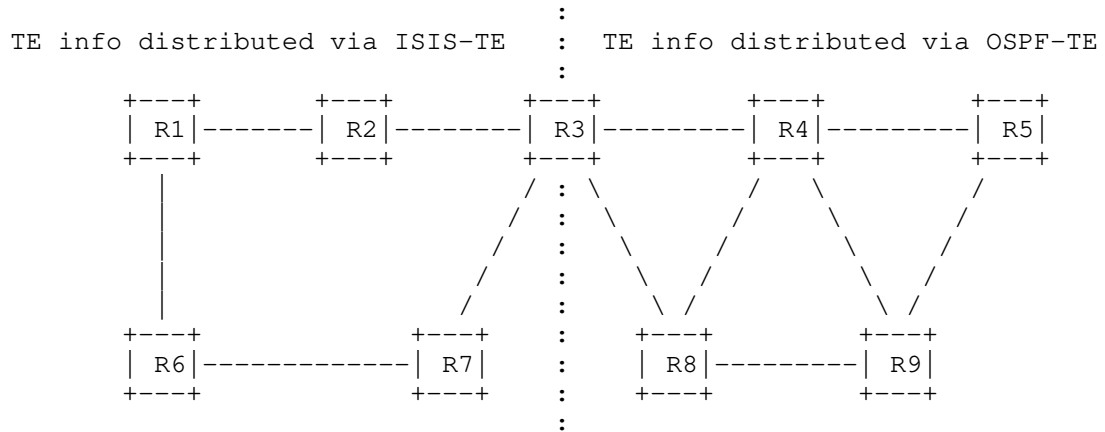


Figure 6a: Example Network Topology

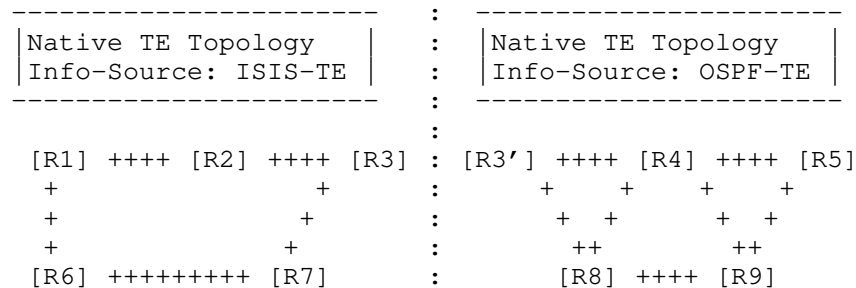


Figure 6b: Native TE Topologies as seen on Node R3

## 4.2. Customized TE Topologies

Customized TE topology is a topology that was modified by the provider to honor a particular client's requirements or preferences. The model discussed in this draft can be used to represent, retrieve and manipulate customized TE Topologies. The model allows the provider to present the network in abstract TE Terms on a per client

basis. These customized topologies contain sufficient information for the path computing client to select paths according to its policies.

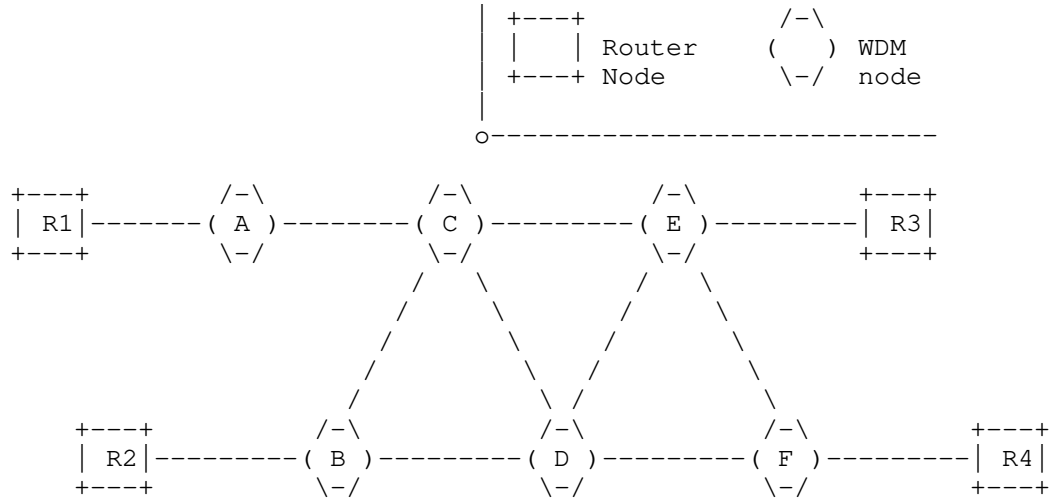


Figure 7: Example packet optical topology

Consider the network topology depicted in Figure 7. This is a typical packet optical transport deployment scenario where the WDM layer network domain serves as a Server Network Domain providing transport connectivity to the packet layer network Domain (Client Network Domain). Nodes R1, R2, R3 and R4 are IP routers that are connected to an Optical WDM transport network. A, B, C, D, E and F are WDM nodes that constitute the Server Network Domain.

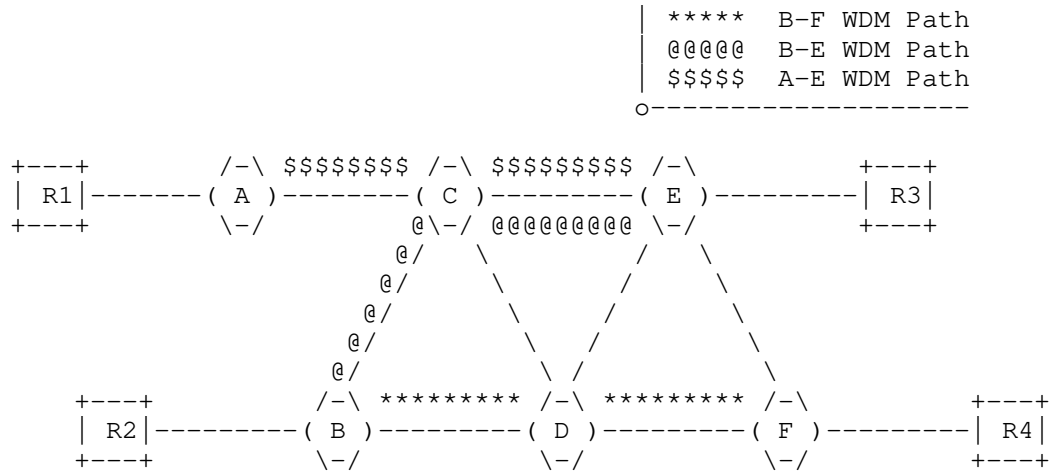


Figure 8a: Paths within the provider domain

```

+++++++ [A] ++++++ [E] ++++++
                ++++++
                ++++++
                ++++++
                ++++++
                ++++++
+++++++ [B] ++++++ [F] ++++++

```

Figure 8b: Customized TE Topology provided to the Client

The goal here is to augment the Client TE Topology with a customized TE Topology provided by the WDM network. Given the availability of the paths A-E, B-F and B-E (Figure 8a), a customized TE Topology as depicted in Figure 8b is provided to the Client. This customized TE Topology is merged with the Client's Native TE Topology and the resulting topology is depicted in Figure 8c.

```

[R1] ++++++ [A] ++++++ [E] ++++++ [R3]
                ++++++
                ++++++
                ++++++
                ++++++
                ++++++
[R2] ++++++ [B] ++++++ [F] ++++++ [R4]

```

Figure 8c: Customized TE Topology merged with the Client's Native TE Topology

The data model proposed in this document can be used to retrieve/represent/manipulate the customized TE Topology depicted in Figure 8b.

A customized TE topology is not necessarily an abstract TE topology. The provider may produce, for example, an abstract TE topology of certain type (e.g. single-abstract-node-with-connectivity-matrix topology, a border-nodes-connected-via-mesh-of-abstract-links topology, etc.) and expose it to all/some clients in expectation that the clients will use it without customization. On the other hand, a client may request a customized version of the provider's native TE topology (e.g. by requesting removal of TE links



which belong to certain layers, are too slow, not protected and/or have a certain affinity). Note that the resulting TE topology will not be abstract (because it will not contain abstract elements), but customized (modified upon client's instructions).

The client ID field in the TE topology identifier (Section 5.4. ) indicates which client the TE topology is customized for. Although an authorized client MAY receive a TE topology with the client ID field matching some other client, the client can customize only TE topologies with the client ID field either 0 or matching the ID of the client in question. If the client starts reconfiguration of a topology its client ID will be automatically set in the topology ID field for all future configurations and updates wrt. the topology in question.

The provider MAY tell the client that a given TE topology cannot be re-negotiated, by setting its own (provider's) ID in the client ID field of the topology ID.

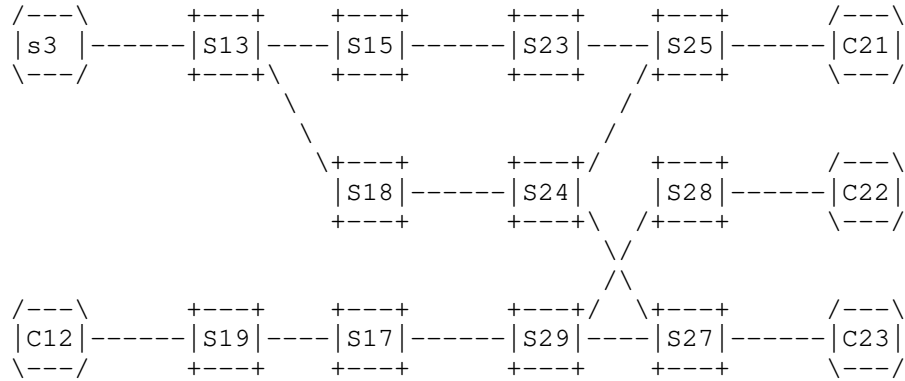
Even though this data model allows to access TE topology information across clients, implementations MAY restrict access for particular clients to particular data fields. The Network Configuration Access Control Model (NACM) [RFC8341] provides such a mechanism.

#### 4.3. Merging TE Topologies Provided by Multiple Providers

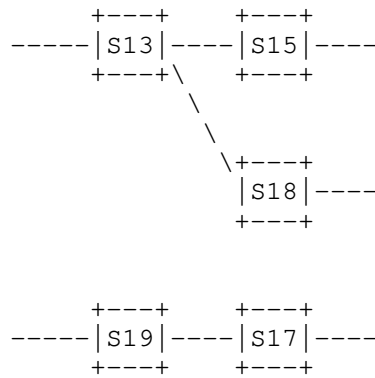
A client may receive TE topologies provided by multiple providers, each of which managing a separate domain of multi-domain network. In order to make use of said topologies, the client is expected to merge the provided TE topologies into one or more client's native TE topologies, each of which homogeneously representing the multi-domain network. This makes it possible for the client to select end-to-end TE paths for its services traversing multiple domains.

In particular, the process of merging TE topologies includes:

- Identifying neighboring domains and locking their topologies horizontally by connecting their inter-domain open-ended TE links;
- Renaming TE node, link, and SRLG IDs to ones allocated from a separate name space; this is necessary because all TE topologies are considered to be, generally speaking, independent with a possibility of clashes among TE node, link or SRLG IDs;
- Locking, vertically, TE topologies associated with different layer networks, according to provided topology inter-layer locks; this is to facilitate inter-layer path computations across multiple TE topologies provided by the same topology provider.



Domain 1 TE Topology



Domain 2 TE Topology

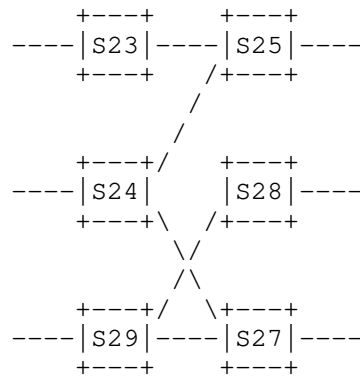


Figure 9: Merging Domain TE Topologies

Figure 9 illustrates the process of merging, by the client, of TE topologies provided by the client's providers. In the Figure, each of the two providers caters to the client (abstract or native) TE topology, describing the network domain under the respective provider's control. The client, by consulting the attributes of the inter-domain TE links - such as inter-domain plug IDs or remote TE node/link IDs (as defined by the TE Topology model) - is able to determine that:

- a) the two domains are adjacent and are inter-connected via three inter-domain TE links, and;

- b) each domain is connected to a separate customer site, connecting the left domain in the Figure to customer devices C-11 and C-12, and the right domain to customer devices C-21, C-22 and C-23.

Therefore, the client inter-connects the open-ended TE links, as shown on the upper part of the Figure.

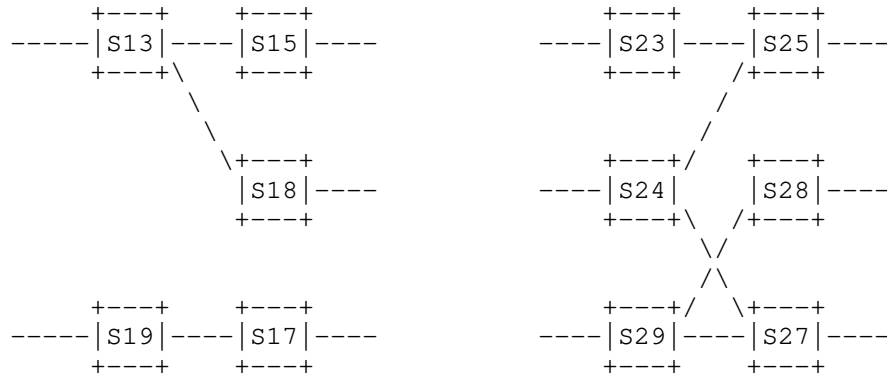
As mentioned, one way to inter-connect the open-ended inter-domain TE links of neighboring domains is to mandate the providers to specify remote nodeID/linkID attribute in the provided inter-domain TE links. This, however, may prove to be not flexible. For example, the providers may not know the respective remote nodeIDs/ linkIDs. More importantly, this option does not allow for the client to mix-n-match multiple (more than one) topologies catered by the same providers (see below). Another, more flexible, option to resolve the open-ended inter-domain TE links is by annotating them with the inter-domain plug ID attribute. Inter-domain plug ID is a network-wide unique number that identifies on the network a connectivity supporting a given inter-domain TE link. Instead of specifying remote node ID/link ID, an inter-domain TE link may provide a non-zero inter-domain plug ID. It is expected that two neighboring domain TE topologies (provided by separate providers) will have each at least one open-ended inter-domain TE link with an inter-domain plug ID matching to one provided by its neighbor. For example, the inter-domain TE link originating from node S15 of the Domain 1 TE topology (Figure 9) and the inter-domain TE link coming from node S23 of Domain 2 TE topology may specify matching inter-domain plug ID (e.g. 175344). This allows for the client to identify adjacent nodes in the separate neighboring TE topologies and resolve the inter-domain TE links connecting them regardless of their respective nodeIDs/linkIDs (which, as mentioned, could be allocated from independent name spaces). Inter-domain plug IDs may be assigned and managed by a central network authority. Alternatively, inter-domain plug IDs could be dynamically auto-discovered (e.g. via LMP protocol).

Furthermore, the client renames the TE nodes, links and SRLGs offered in the abstract TE topologies by assigning to them IDs allocated from a separate name space managed by the client. Such renaming is necessary, because the two abstract TE topologies may have their own name spaces, generally speaking, independent one from another; hence, ID overlaps/clashes are possible. For example, both TE topologies have TE nodes named S7, which, after renaming, appear in the merged TE topology as S17 and S27, respectively.

Once the merging process is complete, the client can use the merged TE topology for path computations across both domains, for example, to compute a TE path connecting C-11 to C-23.

#### 4.4. Dealing with Multiple Abstract TE Topologies Provided by the Same Provider

Domain 1 Abstract TE Topology 1      Domain 2 Abstract TE Topology 1



Domain 1 Abstract TE Topology 1      Domain 2 Abstract TE Topology 1

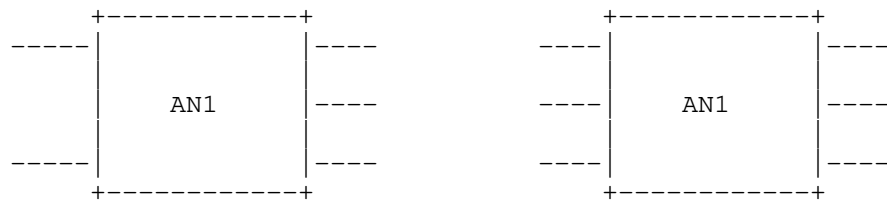


Figure 10: Merging Domain TE Topologies

Based on local configuration, templates and/or policies pushed by the client, a given provider may expose more than one abstract TE topology to the client. For example, one abstract TE topology could be optimized based on a lowest-cost criterion, while another one could be based on best possible delay metrics, while yet another one could be based on maximum bandwidth availability for the client services. Furthermore, the client may request all or some providers to expose additional abstract TE topologies, possibly of a different type and/or optimized differently, as compared to already-provided TE topologies. In any case, the client should be prepared for a provider to offer to the client more than one abstract TE topology.

It should be up to the client (based on the client's local configuration and/or policies conveyed to the client by the client's

clients) to decide how to mix-and-match multiple abstract TE topologies provided by each or some of the providers, as well as how to merge them into the client's native TE topologies. The client also decides how many such merged TE topologies it needs to produce and maintain. For example, in addition to the merged TE topology depicted in the upper part of Figure 9, the client may merge the abstract TE topologies received from the two providers, as shown in Figure 10, into the client's additional native TE topologies, as shown in Figure 11.

Note that allowing for the client mix-n-matching of multiple TE topologies assumes that inter-domain plug IDs (rather than remote nodeID/linkID) option is used for identifying neighboring domains and inter-domain TE link resolution.

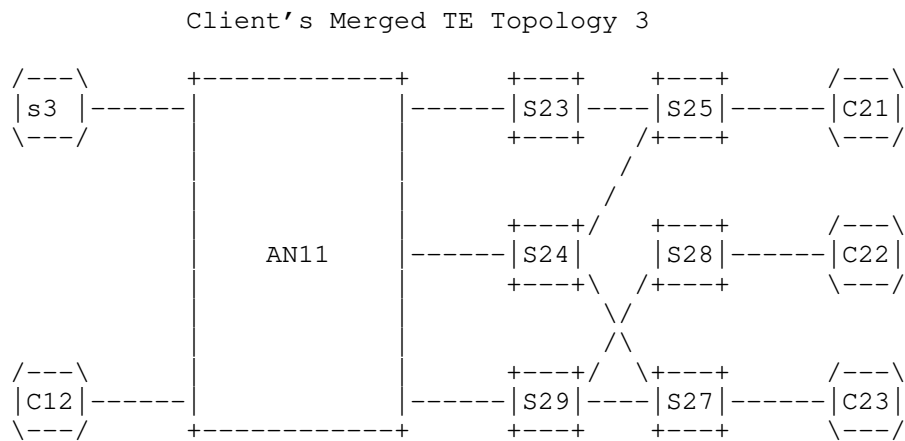
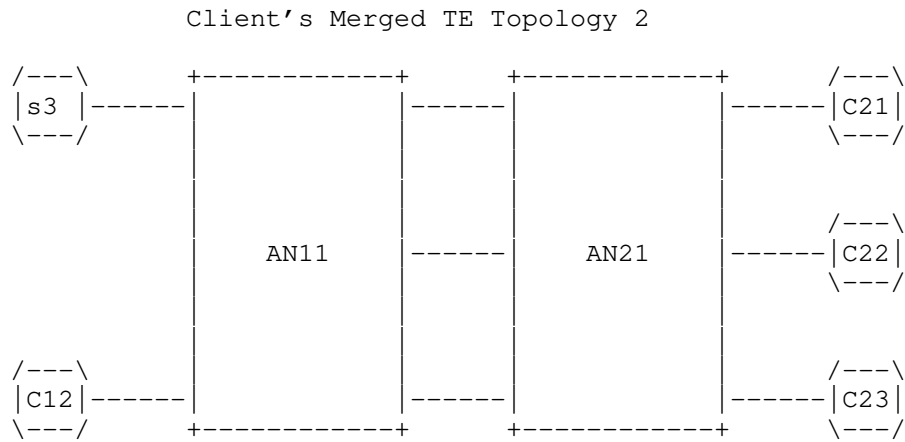


Figure 11: Multiple Native (Merged) Client's TE Topologies

It is important to note that each of the three native (merged) TE topologies could be used by the client for computing TE paths for any of the multi-domain services. The choice as to which topology to use for a given service depends on the service parameters/requirements and the topology's style, optimization criteria and the level of details.

## 5. Modeling Considerations

### 5.1. Network topology building blocks

The network topology building blocks are discussed in [RFC8345]. The TE Topology model proposed in this document augments and uses the `ietf-network-topology` module defined in [RFC8345].

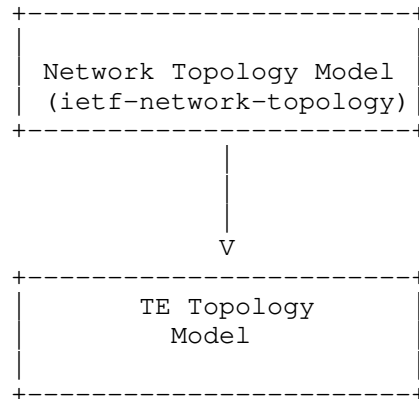


Figure 12: Augmenting the Network Topology Model

### 5.2. Technology agnostic TE Topology model

The TE Topology model proposed in this document is meant to be network technology agnostic. Other technology specific TE Topology models can augment and use the building blocks provided by the proposed model.

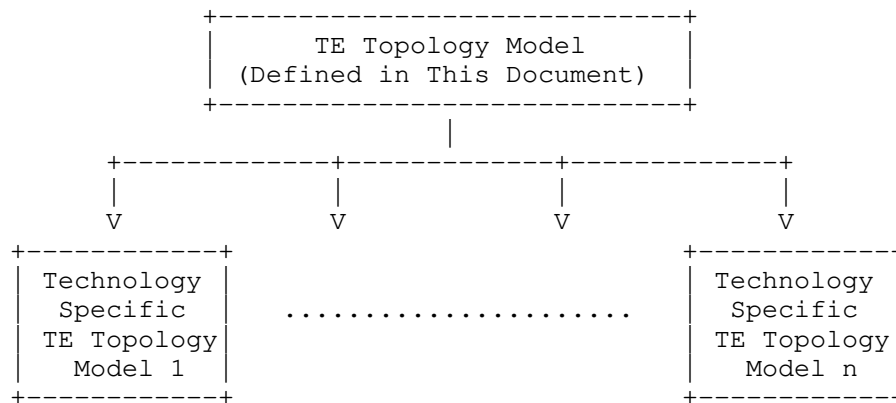


Figure 13: Augmenting the Technology agnostic TE Topology model

### 5.3. Model Structure

The high-level model structure proposed by this document is as shown below:

```

module: ietf-te-topology
augment /nw:networks/nw:network/nw:network-types:
  +--rw te-topology!

augment /nw:networks:
  +--rw te!
  +--rw templates
    +--rw node-template* [name] {template}?
    | .....
    +--rw link-template* [name] {template}?
    .....

augment /nw:networks/nw:network:
  +--rw te-topology-identifier
  | +--rw provider-id?    te-global-id
  | +--rw client-id?     te-global-id
  | +--rw topology-id?   te-topology-id
  +--rw te!
  | .....

augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?     te-types:te-node-id
  +--rw te!
  | .....
  +--rw tunnel-termination-point* [tunnel-tp-id]
  
```



```

      +--rw tunnel-tp-id      binary
      | .....
      +--rw supporting-tunnel-termination-point* [node-ref tunnel-
tp-ref]
      | .....

```

```

augment /nw:networks/nw:network/nt:link:
  +--rw te!
  | .....

```

```

augment /nw:networks/nw:network/nw:node/nt:termination-point:
  +--rw te-tp-id?   te-types:te-tp-id
  +--rw te!
  | .....

```

#### 5.4. Topology Identifiers

The TE-Topology is uniquely identified by a key that has 3 constituents - topology-id, provider-id and client-id. The combination of provider-id and topology-id uniquely identifies a native TE Topology on a given provider. The client-id is used only when Customized TE Topologies come into play; a value of "0" is used as the client-id for native TE Topologies.

```

augment /nw:networks/nw:network:
  +--rw te-topology-identifier
  |   +--rw provider-id?   te-global-id
  |   +--rw client-id?     te-global-id
  |   +--rw topology-id?   te-topology-id
  +--rw te!
  | .....

```

#### 5.5. Generic TE Link Attributes

The model covers the definitions for generic TE Link attributes - bandwidth, admin groups, SRLGs, switching capabilities, TE metric extensions etc.

```

+--rw te-link-attributes
  .....
  +--rw admin-status?                               te-admin-status
  | .....
  +--rw link-index?                                  uint64
  +--rw administrative-group?                         te-types:admin-groups
  +--rw link-protection-type?                         enumeration
  +--rw max-link-bandwidth?                           te-bandwidth

```

```

+--rw max-resv-link-bandwidth?          te-bandwidth
+--rw unreserved-bandwidth* [priority]
|   .....
+--rw te-default-metric?                  uint32
|   .....
+--rw te-srlgs
+--rw te-nsrlgs {nsrlg}?                  .....

```

## 5.6. Generic TE Node Attributes

The model covers the definitions for generic TE Node attributes.

The definition of a generic connectivity matrix is shown below:

```

+--rw te-node-attributes
|   .....
|   +--rw connectivity-matrices
|   |   .....
|   |   +--rw connectivity-matrix* [id]
|   |   |   +--rw id                uint32
|   |   |   +--rw from
|   |   |   |   +--rw tp-ref?         leafref
|   |   |   |   +--rw label-restrictions
|   |   |   +--rw to
|   |   |   |   +--rw tp-ref?         leafref
|   |   |   |   +--rw label-restrictions
|   |   |   +--rw is-allowed?        boolean
|   |   .....
|   |   +--rw underlay! {te-topology-hierarchy}?
|   |   .....
|   |   +--rw path-constraints
|   |   .....
|   |   +--rw optimizations
|   |   .....
|   |   +--rw path-properties
|   |   .....
|   .....

```

The definition of a TTP Local Link Connectivity List is shown below:

```

+--rw tunnel-termination-point* [tunnel-tp-id]
|   +--rw tunnel-tp-id                binary
|   +--rw admin-status?                te-types:te-admin-status
|   +--rw name?                        string
|   +--rw switching-capability?        identityref
|   +--rw encoding?                    identityref
|   +--rw inter-layer-lock-id*         uint32

```

```

    +--rw protection-type?          Identityref
    +--rw client-layer-adaptation
    .....
    +--rw local-link-connectivities
    .....
    |   +--rw local-link-connectivity* [link-tp-ref]
    |       +--rw link-tp-ref          leafref
    |       +--rw label-restrictions
    .....
    |       +--rw is-allowed?          boolean
    |       +--rw underlay {te-topology-hierarchy}?
    .....
    |       +--rw path-constraints
    .....
    |       +--rw optimizations
    .....
    |       +--ro path-properties
    .....
    +--rw supporting-tunnel-termination-point* [node-ref tunnel-tp-
ref]
        +--rw node-ref                inet:uri
        +--rw tunnel-tp-ref            binary

```

The attributes directly under container connectivity-matrices are the default attributes for all connectivity-matrix entries when the per entry corresponding attribute is not specified. When a per entry attribute is specified, it overrides the corresponding attribute directly under the container connectivity-matrices. The same rule applies to the attributes directly under container local-link-connectivities.

Each TTP (Tunnel Termination Point) MAY be supported by one or more supporting TTPs. If the TE node hosting the TTP in question refers to a supporting TE node, then the supporting TTPs are hosted by the supporting TE node. If the TE node refers to an underlay TE topology, the supporting TTPs are hosted by one or more specified TE nodes of the underlay TE topology.

## 5.7. TED Information Sources

The model allows each TE topological element to have multiple TE information sources (OSPF-TE, ISIS-TE, BGP-LS, User-Configured, System-Processed, Other). Each information source is associated with a credibility preference to indicate precedence. In scenarios where a customized TE Topology is merged into a Client's native TE Topology, the merged topological elements would point to the corresponding customized TE Topology as its information source.

```

augment /nw:networks/nw:network/nw:node:
  +--rw te!
  .....
  +--ro information-source?          te-info-source
  +--ro information-source-instance? string
  +--ro information-source-state
  |   +--ro credibility-preference? uint16
  |   +--ro logical-network-element? string
  |   +--ro network-instance?       string
  |   +--ro topology
  |       +--ro node-ref?          leafref
  |       +--ro network-ref?       leafref
  +--ro information-source-entry*
  |   [information-source information-source-instance]
  |   +--ro information-source          te-info-source
  |   +--ro information-source-instance string
  .....

augment /nw:networks/nw:network/nt:link:
  +--rw te!
  .....
  +--ro information-source?          te-info-source
  +--ro information-source-instance? string
  +--ro information-source-state
  |   +--ro credibility-preference? uint16
  |   +--ro logical-network-element? string
  |   +--ro network-instance?       string
  |   +--ro topology
  |       +--ro link-ref?           leafref
  |       +--ro network-ref?        leafref
  +--ro information-source-entry*
  |   [information-source information-source-instance]
  |   +--ro information-source          te-info-source
  |   +--ro information-source-instance string
  .....

```

## 5.8. Overlay/Underlay Relationship

The model captures overlay and underlay relationship for TE nodes/links. For example - in networks where multiple TE Topologies are built hierarchically, this model allows the user to start from a specific topological element in the top most topology and traverse all the way down to the supporting topological elements in the bottom most topology.

This relationship is captured via the "underlay-topology" field for the node and via the "underlay" field for the link. The use of these

fields is optional and this functionality is tagged as a "feature" ("te-topology-hierarchy").

```
augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?   te-types:te-node-id
  +--rw te!
    +--rw te-node-template*          leafref {template}?
    +--rw te-node-attributes
      | .....
      | +--rw underlay-topology {te-topology-hierarchy}?
      |   +--rw network-ref?   leafref
      |
  +--rw te-link-attributes
    | .....
    | +--rw underlay {te-topology-hierarchy}?
    |   +--rw enabled?          boolean
    |   +--rw primary-path
    |     | +--rw network-ref?   leafref
    |     | .....
    |   +--rw backup-path* [index]
    |     | +--rw index          uint32
    |     | +--rw network-ref?   leafref
    |     | .....
    |   +--rw protection-type?    identityref
    |   +--rw tunnel-termination-points
    |     | +--rw source?        binary
    |     | +--rw destination?   binary
    |   +--rw tunnels
    |     | .....
    |     |
```

## 5.9. Templates

The data model provides the users with the ability to define templates and apply them to link and node configurations. The use of "template" configuration is optional and this functionality is tagged as a "feature" ("template").

```
augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?   te-types:te-node-id
  +--rw te!
    +--rw te-node-template*
      | -> ../../../../te/templates/node-template/name
      | {template}?
```

```

augment /nw:networks/nw:network/nt:link:
  +--rw te!
    +--rw te-link-template*
      |      -> ../../../../te/templates/link-template/name
      |      {template}?

augment /nw:networks:
  +--rw te!
    +--rw templates
      +--rw node-template* [name] {template}?
        +--rw name
          |      te-types:te-template-name
        +--rw priority?          uint16
        +--rw reference-change-policy?  enumeration
        +--rw te-node-attributes
        .....
      +--rw link-template* [name] {template}?
        +--rw name
          |      te-types:te-template-name
        +--rw priority?          uint16
        +--rw reference-change-policy?  enumeration
        +--rw te-link-attributes
        .....

```

Multiple templates can be specified to a configuration element. When two or more templates specify values for the same configuration field, the value from the template with the highest priority is used. The range of the priority is from 0 to 65535, with a lower number indicating a higher priority. The reference-change-policy specifies the action that needs to be taken when the template changes on a configuration element that has a reference to this template. The choices of action include taking no action, rejecting the change to the template and applying the change to the corresponding configuration.

#### 5.10. Scheduling Parameters

The model allows time scheduling parameters to be specified for each topological element or for the topology as a whole. These parameters allow the provider to present different topological views to the client at different time slots. The use of "scheduling parameters" is optional.

The YANG data model for configuration scheduling is defined in [I-D.liu-netmod-yang-schedule], which allows specifying configuration schedules without altering this data model.

### 5.11. Notifications

Notifications are a key component of any topology data model.

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

- Subscribe notifications on a per client basis
- Specify subtree filters or xpath filters so that only interested contents will be sent.
- Specify either periodic or on-demand notifications.

### 6. Guidance for Writing Technology Specific TE Topology Augmentations

The TE topology model defined in this document is technology agnostic as it defines concepts, abstractions and attributes that are common across multiple network technologies. It is envisioned that this base model will be widely used when defining technology specific TE topology models for various layer networks.

[I-D.ietf-ccamp-wson-yang], [I-D.ietf-ccamp-otn-topo-yang], and [I-D.ietf-teas-yang-l3-te-topo] are some examples of technology specific TE Topology models. Writers of such models are encouraged to augment the basic TE topology model's containers, such as TE Topology, TE Node, TE Link, Link Termination Point (LTP), Tunnel Termination Point (TTP), Bandwidth and Label with the layer specific attributes instead of defining new containers.

Consider the following technology specific example-topology model:

```
module: example-topology
  augment /nw:networks/nw:network/nw:network-types/tet:te-topology:
    +--rw example-topology!
  augment /nw:networks/nw:network/tet:te:
    +--rw attributes
      +--rw attribute-1?   uint8
  augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes:
      +--rw attributes
        +--rw attribute-2?   uint8
  augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices:
      +--rw attributes
        +--rw attribute-3?   uint8
  augment /nw:networks/nw:network/nw:node/tet:te
```

```

        /tet:te-node-attributes/tet:connectivity-matrices
        /tet:connectivity-matrix:
    +--rw attributes
        +--rw attribute-3?    uint8
    augment /nw:networks/nw:network/nw:node/tet:te
        /tet:tunnel-termination-point:
    +--rw attributes
        +--rw attribute-4?    uint8
    augment /nw:networks/nw:network/nw:node/nt:termination-point
        /tet:te:
    +--rw attributes
        +--rw attribute-5?    uint8
    augment /nw:networks/nw:network/nt:link/tet:te
        /tet:te-link-attributes:
    +--rw attributes
        +--rw attribute-6?    uint8

```

The technology specific TE bandwidth for this example topology can be specified using the following augment statements:

```

    augment /nw:networks/tet:te/tet:templates/tet:link-template
        /tet:te-link-attributes
        /tet:interface-switching-capability/tet:max-lsp-bandwidth
        /tet:te-bandwidth/tet:technology:
    +--:(example)
        +--rw example
            +--rw bandwidth-1?    uint32
    augment /nw:networks/tet:te/tet:templates/tet:link-template
        /tet:te-link-attributes/tet:max-link-bandwidth
        /tet:te-bandwidth/tet:technology:
    +--:(example)
        +--rw example
            +--rw bandwidth-1?    uint32
    augment /nw:networks/tet:te/tet:templates/tet:link-template
        /tet:te-link-attributes/tet:max-resv-link-bandwidth
        /tet:te-bandwidth/tet:technology:
    +--:(example)
        +--rw example
            +--rw bandwidth-1?    uint32
    augment /nw:networks/tet:te/tet:templates/tet:link-template

```



```

        /tet:te-link-attributes/tet:unreserved-bandwidth
        /tet:te-bandwidth/tet:technology:
+--:(example)
  +--rw example
    +--rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:path-constraints/tet:te-bandwidth/tet:technology:
+--:(example)
  +--rw example
    +--rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:path-constraints
  /tet:te-bandwidth/tet:technology:
+--:(example)
  +--rw example
    +--rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:information-source-entry/tet:connectivity-matrices
  /tet:path-constraints/tet:te-bandwidth/tet:technology:
+--:(example)
  +--ro example
    +--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:information-source-entry/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:path-constraints
  /tet:te-bandwidth/tet:technology:
+--:(example)
  +--ro example
    +--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point/tet:client-layer-adaptation
  /tet:switching-capability/tet:te-bandwidth
  /tet:technology:
+--:(example)
  +--rw example
    +--rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities/tet:path-constraints

```

```

        /tet:te-bandwidth/tet:technology:
+---: (example)
+---rw example
+---rw bandwidth-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
        /tet:tunnel-termination-point
        /tet:local-link-connectivities
        /tet:local-link-connectivity/tet:path-constraints
        /tet:te-bandwidth/tet:technology:
+---: (example)
+---rw example
+---rw bandwidth-1?    uint32
augment /nw:networks/nw:network/nt:link/tet:te
        /tet:te-link-attributes
        /tet:interface-switching-capability/tet:max-lsp-bandwidth
        /tet:te-bandwidth/tet:technology:
+---: (example)
+---rw example
+---rw bandwidth-1?    uint32
augment /nw:networks/nw:network/nt:link/tet:te
        /tet:te-link-attributes/tet:max-link-bandwidth
        /tet:te-bandwidth/tet:technology:
+---: (example)
+---rw example
+---rw bandwidth-1?    uint32
augment /nw:networks/nw:network/nt:link/tet:te
        /tet:te-link-attributes/tet:max-resv-link-bandwidth
        /tet:te-bandwidth/tet:technology:
+---: (example)
+---rw example
+---rw bandwidth-1?    uint32
augment /nw:networks/nw:network/nt:link/tet:te
        /tet:information-source-entry
        /tet:interface-switching-capability/tet:max-lsp-bandwidth
        /tet:te-bandwidth/tet:technology:
+---: (example)
+---ro example
+---ro bandwidth-1?    uint32
augment /nw:networks/nw:network/nt:link/tet:te
        /tet:information-source-entry/tet:max-link-bandwidth
        /tet:te-bandwidth/tet:technology:

```

```

+--:(example)
  +--ro example
    +--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry/tet:max-resv-link-bandwidth
  /tet:te-bandwidth/tet:technology:
+--:(example)
  +--ro example
    +--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry/tet:unreserved-bandwidth
  /tet:te-bandwidth/tet:technology:
+--:(example)
  +--ro example
    +--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nw:node/nt:termination-point/tet:te
  /tet:interface-switching-capability/tet:max-lsp-bandwidth
  /tet:te-bandwidth/tet:technology:
+--:(example)
  +--rw example
    +--rw bandwidth-1?  uint32

```

The technology specific TE label for this example topology can be specified using the following augment statements:

```

augment /nw:networks/tet:te/tet:templates/tet:link-template
  /tet:te-link-attributes/tet:underlay/tet:primary-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?  uint32
augment /nw:networks/tet:te/tet:templates/tet:link-template
  /tet:te-link-attributes/tet:underlay/tet:backup-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?  uint32
augment /nw:networks/tet:te/tet:templates/tet:link-template

```

```

        /tet:te-link-attributes/tet:label-restrictions
        /tet:label-restriction/tet:label-start/tet:te-label
        /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/tet:te/tet:templates/tet:link-template
  /tet:te-link-attributes/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-start/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-end/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:underlay/tet:primary-path/tet:path-element/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:underlay/tet:backup-path/tet:path-element/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32

```

```

augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:path-properties/tet:path-route-objects
  /tet:path-route-object/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:from/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:from/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:to/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:to/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te

```

```

        /tet:te-node-attributes/tet:connectivity-matrices
        /tet:connectivity-matrix/tet:underlay/tet:primary-path
        /tet:path-element/tet:type/tet:label/tet:label-hop
        /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:underlay/tet:backup-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:path-properties
  /tet:path-route-objects/tet:path-route-object/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:information-source-entry/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-start/tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:information-source-entry/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-end/tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:information-source-entry/tet:connectivity-matrices
  /tet:underlay/tet:primary-path/tet:path-element/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:

```

```

    +--:(example)
      +--ro example
        +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices
    /tet:underlay/tet:backup-path/tet:path-element/tet:type
    /tet:label/tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices
    /tet:path-properties/tet:path-route-objects
    /tet:path-route-object/tet:type/tet:label/tet:label-hop
    /tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices
    /tet:connectivity-matrix/tet:from/tet:label-restrictions
    /tet:label-restriction/tet:label-start/tet:te-label
    /tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices
    /tet:connectivity-matrix/tet:from/tet:label-restrictions
    /tet:label-restriction/tet:label-end/tet:te-label
    /tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?    uint32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices
    /tet:connectivity-matrix/tet:to/tet:label-restrictions
    /tet:label-restriction/tet:label-start/tet:te-label
    /tet:technology:
+--:(example)
  +--ro example
```

```

      +---ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
      /tet:information-source-entry/tet:connectivity-matrices
      /tet:connectivity-matrix/tet:to/tet:label-restrictions
      /tet:label-restriction/tet:label-end/tet:te-label
      /tet:technology:
+---:(example)
  +---ro example
    +---ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
      /tet:information-source-entry/tet:connectivity-matrices
      /tet:connectivity-matrix/tet:underlay/tet:primary-path
      /tet:path-element/tet:type/tet:label/tet:label-hop
      /tet:te-label/tet:technology:
+---:(example)
  +---ro example
    +---ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
      /tet:information-source-entry/tet:connectivity-matrices
      /tet:connectivity-matrix/tet:underlay/tet:backup-path
      /tet:path-element/tet:type/tet:label/tet:label-hop
      /tet:te-label/tet:technology:
+---:(example)
  +---ro example
    +---ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
      /tet:information-source-entry/tet:connectivity-matrices
      /tet:connectivity-matrix/tet:path-properties
      /tet:path-route-objects/tet:path-route-object/tet:type
      /tet:label/tet:label-hop/tet:te-label/tet:technology:
+---:(example)
  +---ro example
    +---ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
      /tet:tunnel-termination-point
      /tet:local-link-connectivities/tet:label-restrictions
      /tet:label-restriction/tet:label-start/tet:te-label
      /tet:technology:
+---:(example)
  +---rw example
    +---rw label-1?   uint32

```



```
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities/tet:underlay
  /tet:primary-path/tet:path-element/tet:type/tet:label
  /tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities/tet:underlay
  /tet:backup-path/tet:path-element/tet:type/tet:label
  /tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities/tet:path-properties
  /tet:path-route-objects/tet:path-route-object/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?   uint32
```

```

augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
  +--:(example)
    +--rw example
      +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity/tet:underlay
  /tet:primary-path/tet:path-element/tet:type/tet:label
  /tet:label-hop/tet:te-label/tet:technology:
  +--:(example)
    +--rw example
      +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity/tet:underlay/tet:backup-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
  +--:(example)
    +--rw example
      +--rw label-1?   uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:tunnel-termination-point
  /tet:local-link-connectivities
  /tet:local-link-connectivity/tet:path-properties
  /tet:path-route-objects/tet:path-route-object/tet:type
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
  +--:(example)
    +--ro example
      +--ro label-1?   uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
  +--:(example)

```

```

    +--rw example
      +--rw label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes/tet:underlay/tet:primary-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes/tet:underlay/tet:backup-path
  /tet:path-element/tet:type/tet:label/tet:label-hop
  /tet:te-label/tet:technology:
+--:(example)
  +--rw example
    +--rw label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
+--:(example)
  +--ro example
    +--ro label-1?  uint32

```

The YANG module to implement the above example topology can be seen in Appendix C.

## 7. TE Topology YANG Module

This module references [RFC1195], [RFC3209], [RFC3272], [RFC3471], [RFC3630], [RFC3785], [RFC4201], [RFC4202], [RFC4203], [RFC4206], [RFC4872], [RFC5152], [RFC5212], [RFC5305], [RFC5316], [RFC5329], [RFC5392], [RFC6001], [RFC6241], [RFC6991], [RFC7308], [RFC7471], [RFC7579], [RFC7752], [RFC8345], and [I-D.ietf-teas-yang-te-types].

```
<CODE BEGINS> file "ietf-te-topology@2019-02-07.yang"
module ietf-te-topology {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology";

  prefix "tet";

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

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

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

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

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

## organization

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

## contact

"WG Web: <<http://tools.ietf.org/wg/teas/>>

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

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

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

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

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

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

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

## description

"TE topology model for representing and manipulating technology  
agnostic TE Topologies.

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

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

This version of this YANG module is part of RFC XXXX; see the

```
    RFC itself for full legal notices.";

revision "2019-02-07" {
    description "Initial revision";
    reference "RFC XXXX: YANG Data Model for TE Topologies";
    // RFC Ed.: replace XXXX with actual RFC number and remove
    // this note
}

/*
 * Features
 */
feature nsrlg {
    description
        "This feature indicates that the system supports NSRLG
        (Not Sharing Risk Link Group).";
}

feature te-topology-hierarchy {
    description
        "This feature indicates that the system allows underlay
        and/or overlay TE topology hierarchy.";
}

feature template {
    description
        "This feature indicates that the system supports
        template configuration.";
}

/*
 * Typedefs
 */
typedef geographic-coordinate-degree {
    type decimal64 {
        fraction-digits 8;
    }
    description
        "Decimal degree (DD) used to express latitude and longitude
        geographic coordinates.";
} // geographic-coordinate-degree
```

```
typedef te-info-source {
  type enumeration {
    enum "unknown" {
      description "The source is unknown.";
    }
    enum "locally-configured" {
      description "Configured entity.";
    }
    enum "ospfv2" {
      description "OSPFv2.";
    }
    enum "ospfv3" {
      description "OSPFv3.";
    }
    enum "isis" {
      description "ISIS.";
    }
    enum "bgp-ls" {
      description "BGP-LS.";
      reference
        "RFC 7752: North-Bound Distribution of Link-State and
        Traffic Engineering (TE) Information Using BGP";
    }
    enum "system-processed" {
      description "System processed entity.";
    }
    enum "other" {
      description "Other source.";
    }
  }
  description
    "Describing the type of source that has provided the
    related information, and the source credibility.";
} // te-info-source

/*
 * Groupings
 */
grouping connectivity-matrix-entry-path-attributes {
  description
```

```
    "Attributes of connectivity matrix entry.";
  leaf is-allowed {
    type boolean;
    description
      "true - switching is allowed,
       false - switching is disallowed.";
  }
  container underlay {
    if-feature te-topology-hierarchy;
    description "Attributes of the te-link underlay.";
    reference
      "RFC 4206: Label Switched Paths (LSP) Hierarchy with
       Generalized Multi-Protocol Label Switching (GMPLS)
       Traffic Engineering (TE)";

    uses te-link-underlay-attributes;
  } // underlay

  uses te-types:generic-path-constraints;
  uses te-types:generic-path-optimization;
  uses te-types:generic-path-properties;
} // connectivity-matrix-entry-path-attributes

grouping geolocation-container {
  description
    "A container containing a GPS location.";
  container geolocation{
    config false;
    description
      "A container containing a GPS location.";
    leaf altitude {
      type int64;
      units millimeter;
      description
        "Distance above the sea level.";
    }
    leaf latitude {
      type geographic-coordinate-degree {
        range "-90..90";
      }
      description

```



```
        "Relative position north or south on the Earth's surface.";
    }
    leaf longitude {
        type geographic-coordinate-degree {
            range "-180..180";
        }
        description
            "Angular distance east or west on the Earth's surface.";
    }
} // gps-location
} // geolocation-container

grouping information-source-state-attributes {
    description
        "The attributes identifying source that has provided the
        related information, and the source credibility.";
    leaf credibility-preference {
        type uint16;
        description
            "The preference value to calculate the traffic
            engineering database credibility value used for
            tie-break selection between different
            information-source values.
            Higher value is more preferable.";
    }
    leaf logical-network-element {
        type string;
        description
            "When applicable, this is the name of a logical network
            element from which the information is learned.";
    } // logical-network-element
    leaf network-instance {
        type string;
        description
            "When applicable, this is the name of a network-instance
            from which the information is learned.";
    } // network-instance
} // information-source-state-attributes

grouping information-source-per-link-attributes {
    description
```

```
    "Per node container of the attributes identifying source that
      has provided the related information, and the source
      credibility.";
  leaf information-source {
    type te-info-source;
    config false;
    description
      "Indicates the type of the information source.";
  }
  leaf information-source-instance {
    type string;
    config false;
    description
      "The name indicating the instance of the information
        source.";
  }
  container information-source-state {
    config false;
    description
      "The container contains state attributes related to
        the information source.";
    uses information-source-state-attributes;
    container topology {
      description
        "When the information is processed by the system,
          the attributes in this container indicate which topology
          is used to process to generate the result information.";
      uses nt:link-ref;
    } // topology
  } // information-source-state
} // information-source-per-link-attributes

grouping information-source-per-node-attributes {
  description
    "Per node container of the attributes identifying source that
      has provided the related information, and the source
      credibility.";
  leaf information-source {
    type te-info-source;
    config false;
    description
```

```
        "Indicates the type of the information source.";
    }
    leaf information-source-instance {
        type string;
        config false;
        description
            "The name indicating the instance of the information
            source.";
    }
    container information-source-state {
        config false;
        description
            "The container contains state attributes related to
            the information source.";
        uses information-source-state-attributes;
        container topology {
            description
                "When the information is processed by the system,
                the attributes in this container indicate which topology
                is used to process to generate the result information.";
            uses nw:node-ref;
        } // topology
    } // information-source-state
} // information-source-per-node-attributes

grouping interface-switching-capability-list {
    description
        "List of Interface Switching Capabilities Descriptors (ISCD)";
    list interface-switching-capability {
        key "switching-capability encoding";
        description
            "List of Interface Switching Capabilities Descriptors (ISCD)
            for this link.";
        reference
            "RFC 3471: Generalized Multi-Protocol Label Switching (GMPLS)
            Signaling Functional Description.
            RFC 4203: OSPF Extensions in Support of Generalized
            Multi-Protocol Label Switching (GMPLS).";
    }
    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.";
  }
  uses te-link-iscd-attributes;
} // interface-switching-capability
} // interface-switching-capability-list

grouping statistics-per-link {
  description
    "Statistics attributes per TE link.";
  leaf discontinuity-time {
    type yang:date-and-time;
    description
      "The time on the most recent occasion at which any one or
      more of this interface's counters suffered a
      discontinuity. If no such discontinuities have occurred
      since the last re-initialization of the local management
      subsystem, then this node contains the time the local
      management subsystem re-initialized itself.";
  }
  /* Administrative attributes */
  leaf disables {
    type yang:counter32;
    description
      "Number of times that link was disabled.";
  }
  leaf enables {
    type yang:counter32;
    description
      "Number of times that link was enabled.";
  }
  leaf maintenance-clears {
    type yang:counter32;
```

```
        description
            "Number of times that link was put out of maintenance.";
    }
    leaf maintenance-sets {
        type yang:counter32;
        description
            "Number of times that link was put in maintenance.";
    }
    leaf modifies {
        type yang:counter32;
        description
            "Number of times that link was modified.";
    }
    /* Operational attributes */
    leaf downs {
        type yang:counter32;
        description
            "Number of times that link was set to operational down.";
    }
    leaf ups {
        type yang:counter32;
        description
            "Number of times that link was set to operational up.";
    }
    /* Recovery attributes */
    leaf fault-clears {
        type yang:counter32;
        description
            "Number of times that link experienced fault clear event.";
    }
    leaf fault-detects {
        type yang:counter32;
        description
            "Number of times that link experienced fault detection.";
    }
    leaf protection-switches {
        type yang:counter32;
        description
            "Number of times that link experienced protection
            switchover.";
    }
}
```

```
leaf protection-reverts {
  type yang:counter32;
  description
    "Number of times that link experienced protection
    reversion.";
}
leaf restoration-failures {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    failure.";
}
leaf restoration-starts {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    start.";
}
leaf restoration-successes {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    success.";
}
leaf restoration-reversion-failures {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    failure.";
}
leaf restoration-reversion-starts {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    start.";
}
leaf restoration-reversion-successes {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    success.";
```

```
    }  
  } // statistics-per-link  
  
  grouping statistics-per-node {  
    description  
      "Statistics attributes per TE node.";  
    leaf discontinuity-time {  
      type yang:date-and-time;  
      description  
        "The time on the most recent occasion at which any one or  
        more of this interface's counters suffered a  
        discontinuity. If no such discontinuities have occurred  
        since the last re-initialization of the local management  
        subsystem, then this node contains the time the local  
        management subsystem re-initialized itself.";  
    }  
    container node {  
      description  
        "Containing TE node level statistics attributes.";  
      leaf disables {  
        type yang:counter32;  
        description  
          "Number of times that node was disabled.";  
      }  
      leaf enables {  
        type yang:counter32;  
        description  
          "Number of times that node was enabled.";  
      }  
      leaf maintenance-sets {  
        type yang:counter32;  
        description  
          "Number of times that node was put in maintenance.";  
      }  
      leaf maintenance-clears {  
        type yang:counter32;  
        description  
          "Number of times that node was put out of maintenance.";  
      }  
      leaf modifies {  
        type yang:counter32;
```

```
        description
            "Number of times that node was modified.";
    }
} // node
container connectivity-matrix-entry {
    description
        "Containing connectivity matrix entry level statistics
        attributes.";
    leaf creates {
        type yang:counter32;
        description
            "Number of times that a connectivity matrix entry was
            created.";
        reference
            "RFC 6241. Section 7.2 for 'create' operation. ";
    }
    leaf deletes {
        type yang:counter32;
        description
            "Number of times that a connectivity matrix entry was
            deleted.";
        reference
            "RFC 6241. Section 7.2 for 'delete' operation. ";
    }
    leaf disables {
        type yang:counter32;
        description
            "Number of times that a connectivity matrix entry was
            disabled.";
    }
    leaf enables {
        type yang:counter32;
        description
            "Number of times that a connectivity matrix entry was
            enabled.";
    }
    leaf modifies {
        type yang:counter32;
        description
            "Number of times that a connectivity matrix entry was
            modified.";
```



```
    }
  } // connectivity-matrix-entry
} // statistics-per-node

grouping statistics-per-ttp {
  description
    "Statistics attributes per TE TTP (Tunnel Termination Point).";
  leaf discontinuity-time {
    type yang:date-and-time;
    description
      "The time on the most recent occasion at which any one or
      more of this interface's counters suffered a
      discontinuity.  If no such discontinuities have occurred
      since the last re-initialization of the local management
      subsystem, then this node contains the time the local
      management subsystem re-initialized itself.";
  }
  container tunnel-termination-point {
    description
      "Containing TE TTP (Tunnel Termination Point) level
      statistics attributes.";
    /* Administrative attributes */
    leaf disables {
      type yang:counter32;
      description
        "Number of times that TTP was disabled.";
    }
    leaf enables {
      type yang:counter32;
      description
        "Number of times that TTP was enabled.";
    }
    leaf maintenance-clears {
      type yang:counter32;
      description
        "Number of times that TTP was put out of maintenance.";
    }
    leaf maintenance-sets {
      type yang:counter32;
      description
        "Number of times that TTP was put in maintenance.";
    }
  }
}
```

```
    }
    leaf modifies {
        type yang:counter32;
        description
            "Number of times that TTP was modified.";
    }
    /* Operational attributes */
    leaf downs {
        type yang:counter32;
        description
            "Number of times that TTP was set to operational down.";
    }
    leaf ups {
        type yang:counter32;
        description
            "Number of times that TTP was set to operational up.";
    }
    leaf in-service-clears {
        type yang:counter32;
        description
            "Number of times that TTP was taken out of service
            (TE tunnel was released).";
    }
    leaf in-service-sets {
        type yang:counter32;
        description
            "Number of times that TTP was put in service by a TE
            tunnel (TE tunnel was set up).";
    }
} // tunnel-termination-point

container local-link-connectivity {
    description
        "Containing TE LLCL (Local Link Connectivity List) level
        statistics attributes.";
    leaf creates {
        type yang:counter32;
        description
            "Number of times that an LLCL entry was created.";
        reference
            "RFC 6241. Section 7.2 for 'create' operation.";
    }
}
```

```
    }
    leaf deletes {
      type yang:counter32;
      description
        "Number of times that an LLCL entry was deleted.";
      reference
        "RFC 6241. Section 7.2 for 'delete' operation.";
    }
    leaf disables {
      type yang:counter32;
      description
        "Number of times that an LLCL entry was disabled.";
    }
    leaf enables {
      type yang:counter32;
      description
        "Number of times that an LLCL entry was enabled.";
    }
    leaf modifies {
      type yang:counter32;
      description
        "Number of times that an LLCL entry was modified.";
    }
  } // local-link-connectivity
} // statistics-per-ttp

grouping te-link-augment {
  description
    "Augmentation for TE link.";
  uses te-link-config;
  uses te-link-state-derived;
  container statistics {
    config false;
    description
      "Statistics data.";
    uses statistics-per-link;
  } // statistics
} // te-link-augment

grouping te-link-config {
  description
```

```
"TE link configuration grouping.";
choice bundle-stack-level {
  description
    "The TE link can be partitioned into bundled
    links, or component links.";
  case bundle {
    container bundled-links {
      description
        "A set of bundled links.";
      reference
        "RFC 4201: Link Bundling in MPLS Traffic Engineering
        (TE).";
      list bundled-link {
        key "sequence";
        description
          "Specify a bundled interface that is
          further partitioned.";
        leaf sequence {
          type uint32;
          description
            "Identify the sequence in the bundle.";
        }
      } // list bundled-link
    }
  }
  case component {
    container component-links {
      description
        "A set of component links";
      list component-link {
        key "sequence";
        description
          "Specify a component interface that is
          sufficient to unambiguously identify the
          appropriate resources";

        leaf sequence {
          type uint32;
          description
            "Identify the sequence in the bundle.";
        }
      }
    }
  }
}
```

```
        leaf src-interface-ref {
            type string;
            description
                "Reference to component link interface on the
                 source node.";
        }
        leaf des-interface-ref {
            type string;
            description
                "Reference to component link interface on the
                 destination node.";
        }
    }
}
} // bundle-stack-level

leaf-list te-link-template {
    if-feature template;
    type leafref {
        path "../.../te/templates/link-template/name";
    }
    description
        "The reference to a TE link template.";
}
uses te-link-config-attributes;
} // te-link-config

grouping te-link-config-attributes {
    description
        "Link configuration attributes in a TE topology.";
    container te-link-attributes {
        description "Link attributes in a TE topology.";
        leaf access-type {
            type te-types:te-link-access-type;
            description
                "Link access type, which can be point-to-point or
                 multi-access.";
        }
        container external-domain {
            description
```

```
    "For an inter-domain link, specify the attributes of
      the remote end of link, to facilitate the signalling at
      local end.";
  uses nw:network-ref;
  leaf remote-te-node-id {
    type te-types:te-node-id;
    description
      "Remote TE node identifier, used together with
       remote-te-link-id to identify the remote link
       termination point in a different domain.";
  }
  leaf remote-te-link-tp-id {
    type te-types:te-tp-id;
    description
      "Remote TE link termination point identifier, used
       together with remote-te-node-id to identify the remote
       link termination point in a different domain.";
  }
}
leaf is-abstract {
  type empty;
  description "Present if the link is abstract.";
}
leaf name {
  type string;
  description "Link Name.";
}
container underlay {
  if-feature te-topology-hierarchy;
  description "Attributes of the te-link underlay.";
  reference
    "RFC 4206: Label Switched Paths (LSP) Hierarchy with
     Generalized Multi-Protocol Label Switching (GMPLS)
     Traffic Engineering (TE)";

  uses te-link-underlay-attributes;
} // underlay
leaf admin-status {
  type te-types:te-admin-status;
  description
    "The administrative state of the link.";
```

```
    }

    uses te-link-info-attributes;
  } // te-link-attributes
} // te-link-config-attributes

grouping te-link-info-attributes {
  description
    "Advertised TE information attributes.";
  leaf link-index {
    type uint64;
    description
      "The link identifier.  If OSPF is used, this represents an
      ospfLsdbID.  If IS-IS is used, this represents an isisLSPID.
      If a locally configured link is used, this object represents
      a unique value, which is locally defined in a router.";
  }
  leaf administrative-group {
    type te-types:admin-groups;
    description
      "Administrative group or color of the link.
      This attribute covers both administrative group (defined in
      RFC 3630, RFC 5305 and RFC 5329), and extended
      administrative group (defined in RFC 7308).";
  }
}

uses interface-switching-capability-list;
uses te-types:label-set-info;

leaf link-protection-type {
  type identityref {
    base te-types:link-protection-type;
  }
  description
    "Link Protection Type desired for this link.";
  reference
    "RFC 4202: Routing Extensions in Support of
    Generalized Multi-Protocol Label Switching (GMPLS).";
}

container max-link-bandwidth {
```

```
    uses te-types:te-bandwidth;
    description
        "Maximum bandwidth that can be seen on this link in this
        direction. Units in bytes per second.";
    reference
        "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
        Version 2.
        RFC 5305: IS-IS Extensions for Traffic Engineering.";
}
container max-resv-link-bandwidth {
    uses te-types:te-bandwidth;
    description
        "Maximum amount of bandwidth that can be reserved in this
        direction in this link. Units in bytes per second.";
    reference
        "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
        Version 2.
        RFC 5305: IS-IS Extensions for Traffic Engineering.";
}
list unreserved-bandwidth {
    key "priority";
    max-elements "8";
    description
        "Unreserved bandwidth for 0-7 priority levels. Units in
        bytes per second.";
    reference
        "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
        Version 2.
        RFC 5305: IS-IS Extensions for Traffic Engineering.";
    leaf priority {
        type uint8 {
            range "0..7";
        }
        description "Priority.";
    }
    uses te-types:te-bandwidth;
}
leaf te-default-metric {
    type uint32;
    description
        "Traffic engineering metric.";
```



```
    reference
      "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
      Version 2.
      RFC 5305: IS-IS Extensions for Traffic Engineering.";
  }
  leaf te-delay-metric {
    type uint32;
    description
      "Traffic engineering delay metric.";
    reference
      "RFC 7471: OSPF Traffic Engineering (TE) Metric Extensions.";
  }
  leaf te-igp-metric {
    type uint32;
    description
      "IGP metric used for traffic engineering.";
    reference
      "RFC 3785: Use of Interior Gateway Protocol (IGP) Metric as a
      Second MPLS Traffic Engineering (TE) Metric.";
  }
  container te-srlgs {
    description
      "Containing a list of SLRGs.";
    leaf-list value {
      type te-types:srlg;
      description "SRLG value.";
      reference
        "RFC 4202: Routing Extensions in Support of
        Generalized Multi-Protocol Label Switching (GMPLS).";
    }
  }
  container te-nsrlgs {
    if-feature nsrlg;
    description
      "Containing a list of NSRLGs (Not Sharing Risk Link
      Groups).
      When an abstract TE link is configured, this list specifies
      the request that underlay TE paths need to be mutually
      disjoint with other TE links in the same groups.";
    leaf-list id {
      type uint32;
```

```
        description
            "NSRLG ID, uniquely configured within a topology.";
        reference
            "RFC 4872: RSVP-TE Extensions in Support of End-to-End
            Generalized Multi-Protocol Label Switching (GMPLS)
            Recovery";
    }
}
} // te-link-info-attributes

grouping te-link-iscd-attributes {
    description
        "TE link ISCD (Interface Switching Capability Descriptor)
        attributes.";
    reference
        "Sec 1.4, RFC 4203: OSPF Extensions in Support of Generalized
        Multi-Protocol Label Switching (GMPLS). Section 1.4.";
    list max-lsp-bandwidth {
        key "priority";
        max-elements "8";
        description
            "Maximum LSP Bandwidth at priorities 0-7.";
        leaf priority {
            type uint8 {
                range "0..7";
            }
            description "Priority.";
        }
        uses te-types:te-bandwidth;
    }
} // te-link-iscd-attributes

grouping te-link-state-derived {
    description
        "Link state attributes in a TE topology.";
    leaf oper-status {
        type te-types:te-oper-status;
        config false;
        description
            "The current operational state of the link.";
    }
}
```

```
leaf is-transitional {
  type empty;
  config false;
  description
    "Present if the link is transitional, used as an
     alternative approach in lieu of inter-layer-lock-id
     for path computation in a TE topology covering multiple
     layers or multiple regions.";
  reference
    "RFC 5212: Requirements for GMPLS-Based Multi-Region and
     Multi-Layer Networks (MRN/MLN).
     RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
     for Multi-Layer and Multi-Region Networks (MLN/MRN).";
}
uses information-source-per-link-attributes;
list information-source-entry {
  key "information-source information-source-instance";
  config false;
  description
    "A list of information sources learned, including the one
     used.";
  uses information-source-per-link-attributes;
  uses te-link-info-attributes;
}
container recovery {
  config false;
  description
    "Status of the recovery process.";
  leaf restoration-status {
    type te-types:te-recovery-status;
    description
      "Restoration status.";
  }
  leaf protection-status {
    type te-types:te-recovery-status;
    description
      "Protection status.";
  }
}
container underlay {
  if-feature te-topology-hierarchy;
```

```
    config false;
    description "State attributes for te-link underlay.";
    leaf dynamic {
        type boolean;
        description
            "true if the underlay is dynamically created.";
    }
    leaf committed {
        type boolean;
        description
            "true if the underlay is committed.";
    }
}
} // te-link-state-derived

grouping te-link-underlay-attributes {
    description "Attributes for te-link underlay.";
    reference
        "RFC 4206: Label Switched Paths (LSP) Hierarchy with
        Generalized Multi-Protocol Label Switching (GMPLS)
        Traffic Engineering (TE)";
    leaf enabled {
        type boolean;
        description
            "'true' if the underlay is enabled.
            'false' if the underlay is disabled.";
    }
}
container primary-path {
    description
        "The service path on the underlay topology that
        supports this link.";
    uses nw:network-ref;
    list path-element {
        key "path-element-id";
        description
            "A list of path elements describing the service path.";
        leaf path-element-id {
            type uint32;
            description "To identify the element in a path.";
        }
    }
    uses te-path-element;
```

```
    }  
  } // primary-path  
  list backup-path {  
    key "index";  
    description  
      "A list of backup service paths on the underlay topology that  
      protect the underlay primary path. If the primary path is  
      not protected, the list contains zero elements. If the  
      primary path is protected, the list contains one or more  
      elements.";  
    leaf index {  
      type uint32;  
      description  
        "A sequence number to identify a backup path.";  
    }  
    uses nw:network-ref;  
    list path-element {  
      key "path-element-id";  
      description  
        "A list of path elements describing the backup service  
        path";  
      leaf path-element-id {  
        type uint32;  
        description "To identify the element in a path.";  
      }  
      uses te-path-element;  
    }  
  } // underlay-backup-path  
  leaf protection-type {  
    type identityref {  
      base te-types:lsp-protection-type;  
    }  
    description  
      "Underlay protection type desired for this link.";  
  }  
  container tunnel-termination-points {  
    description  
      "Underlay TTP (Tunnel Termination Points) desired for this  
      link.";  
    leaf source {  
      type binary;  
    }  
  }  
}
```

```
        description
            "Source tunnel termination point identifier.";
    }
    leaf destination {
        type binary;
        description
            "Destination tunnel termination point identifier.";
    }
}
container tunnels {
    description
        "Underlay TE tunnels supporting this TE link.";
    leaf sharing {
        type boolean;
        default true;
        description
            "'true' if the underlay tunnel can be shared with other
            TE links;
            'false' if the underlay tunnel is dedicated to this
            TE link.
            This leaf is the default option for all TE tunnels,
            and may be overridden by the per TE tunnel value.";
    }
    list tunnel {
        key "tunnel-name";
        description
            "Zero, one or more underlay TE tunnels that support this TE
            link.";
        leaf tunnel-name {
            type string;
            description
                "A tunnel name uniquely identifies an underlay TE tunnel,
                used together with the source-node of this link.
                The detailed information of this tunnel can be retrieved
                from the ietf-te model.";
            reference "RFC 3209";
        }
    }
    leaf sharing {
        type boolean;
        description
            "'true' if the underlay tunnel can be shared with other
```

```
        TE links;
        'false' if the underlay tunnel is dedicated to this
        TE link.";
    }
} // tunnel
} // tunnels
} // te-link-underlay-attributes

grouping te-node-augment {
  description
    "Augmentation for TE node.";
  uses te-node-config;
  uses te-node-state-derived;
  container statistics {
    config false;
    description
      "Statistics data.";
    uses statistics-per-node;
  } // statistics

  list tunnel-termination-point {
    key "tunnel-tp-id";
    description
      "A termination point can terminate a tunnel.";
    leaf tunnel-tp-id {
      type binary;
      description
        "Tunnel termination point identifier.";
    }

    uses te-node-tunnel-termination-point-config;
    leaf oper-status {
      type te-types:te-oper-status;
      config false;
      description
        "The current operational state of the tunnel
        termination point.";
    }
    uses geolocation-container;
    container statistics {
      config false;
```

```
    description
      "Statistics data.";
    uses statistics-per-ttp;
  } // statistics

  // Relations to other tunnel termination points
  list supporting-tunnel-termination-point {
    key "node-ref tunnel-tp-ref";
    description
      "Identifies the tunnel termination points, that this
       tunnel termination point is depending on.";
    leaf node-ref {
      type inet:uri;
      description
        "This leaf identifies the node in which the supporting
         tunnel termination point is present.
         This node is either the supporting node or a node in an
         underlay topology.";
    }
    leaf tunnel-tp-ref {
      type binary;
      description
        "Reference to a tunnel termination point, which is
         either in the supporting node or a node in an
         underlay topology.";
    }
  } // supporting-tunnel-termination-point
} // tunnel-termination-point
} // te-node-augment

grouping te-node-config {
  description "TE node configuration grouping.";
  leaf-list te-node-template {
    if-feature template;
    type leafref {
      path "../.../.../te/templates/node-template/name";
    }
    description
      "The reference to a TE node template.";
  }
  uses te-node-config-attributes;
```



```
    } // te-node-config

    grouping te-node-config-attributes {
      description "Configuration node attributes in a TE topology.";
      container te-node-attributes {
        description "Containing node attributes in a TE topology.";
        leaf admin-status {
          type te-types:te-admin-status;
          description
            "The administrative state of the link.";
        }
        uses te-node-connectivity-matrices;
        uses te-node-info-attributes;
      } // te-node-attributes
    } // te-node-config-attributes

    grouping te-node-config-attributes-template {
      description
        "Configuration node attributes for template in a TE topology.";
      container te-node-attributes {
        description "Containing node attributes in a TE topology.";
        leaf admin-status {
          type te-types:te-admin-status;
          description
            "The administrative state of the link.";
        }
        uses te-node-info-attributes;
      } // te-node-attributes
    } // te-node-config-attributes-template

    grouping te-node-connectivity-matrices {
      description "Connectivity matrix on a TE node.";
      container connectivity-matrices {
        description
          "Containing connectivity matrix on a TE node.";
        leaf number-of-entries {
          type uint16;
          description
            "The number of connectivity matrix entries.
             If this number is specified in the configuration request,
             the number is requested number of entries, which may not
```

```
        all be listed in the list;
        if this number is reported in the state data,
        the number is the current number of operational entries.";
    }
    uses te-types:label-set-info;
    uses connectivity-matrix-entry-path-attributes;
    list connectivity-matrix {
        key "id";
        description
            "Represents node's switching limitations, i.e. limitations
            in interconnecting network TE links across the node.";
        reference
            "RFC 7579: General Network Element Constraint Encoding
            for GMPLS-Controlled Networks.";
        leaf id {
            type uint32;
            description "Identifies the connectivity-matrix entry.";
        }
    } // connectivity-matrix
} // connectivity-matrices
} // te-node-connectivity-matrices

grouping te-node-connectivity-matrix-attributes {
    description
        "Termination point references of a connectivity matrix entry.";
    container from {
        description
            "Reference to source link termination point.";
        leaf tp-ref {
            type leafref {
                path "../..//../..//../nt:termination-point/nt:tp-id";
            }
            description
                "Relative reference to a termination point.";
        }
        uses te-types:label-set-info;
    }
    container to {
        description
            "Reference to destination link termination point.";
        leaf tp-ref {
```

```
    type leafref {
      path "../.../.../.../nt:termination-point/nt:tp-id";
    }
    description
      "Relative reference to a termination point.";
  }
  uses te-types:label-set-info;
}
uses connectivity-matrix-entry-path-attributes;
} // te-node-connectivity-matrix-attributes

grouping te-node-info-attributes {
  description
    "Advertised TE information attributes.";
  leaf domain-id {
    type uint32;
    description
      "Identifies the domain that this node belongs.
       This attribute is used to support inter-domain links.";
    reference
      "RFC 5152: A Per-Domain Path Computation Method for
       Establishing Inter-Domain Traffic Engineering (TE)
       Label Switched Paths (LSPs).
       RFC 5392: OSPF Extensions in Support of Inter-Autonomous
       System (AS) MPLS and GMPLS Traffic Engineering.
       RFC 5316: ISIS Extensions in Support of Inter-Autonomous
       System (AS) MPLS and GMPLS Traffic Engineering.";
  }
  leaf is-abstract {
    type empty;
    description
      "Present if the node is abstract, not present if the node
       is actual.";
  }
  leaf name {
    type string;
    description "Node name.";
  }
  leaf-list signaling-address {
    type inet:ip-address;
    description "Node signaling address.";
  }
}
```

```
    }
    container underlay-topology {
      if-feature te-topology-hierarchy;
      description
        "When an abstract node encapsulates a topology,
         the attributes in this container point to said topology.";
      uses nw:network-ref;
    }
  } // te-node-info-attributes

grouping te-node-state-derived {
  description "Node state attributes in a TE topology.";
  leaf oper-status {
    type te-types:te-oper-status;
    config false;
    description
      "The current operational state of the node.";
  }
  uses geolocation-container;
  leaf is-multi-access-dr {
    type empty;
    config false;
    description
      "The presence of this attribute indicates that this TE node
       is a pseudonode elected as a designated router.";
    reference
      "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
       Version 2.
       RFC 1195: Use of OSI IS-IS for Routing in TCP/IP and Dual
       Environments.";
  }
  uses information-source-per-node-attributes;
  list information-source-entry {
    key "information-source information-source-instance";
    config false;
    description
      "A list of information sources learned, including the one
       used.";
    uses information-source-per-node-attributes;
    uses te-node-connectivity-matrices;
    uses te-node-info-attributes;
  }
}
```

```
    }  
  } // te-node-state-derived  
  
  grouping te-node-tunnel-termination-point-config {  
    description  
      "Termination capability of a tunnel termination point on a  
      TE node.";  
    uses te-node-tunnel-termination-point-config-attributes;  
    container local-link-connectivities {  
      description  
        "Containing local link connectivity list for  
        a tunnel termination point on a TE node.";  
      leaf number-of-entries {  
        type uint16;  
        description  
          "The number of local link connectivity list entries.  
          If this number is specified in the configuration request,  
          the number is requested number of entries, which may not  
          all be listed in the list;  
          if this number is reported in the state data,  
          the number is the current number of operational entries.";  
        }  
      uses te-types:label-set-info;  
      uses connectivity-matrix-entry-path-attributes;  
    } // local-link-connectivities  
  } // te-node-tunnel-termination-point-config  
  
  grouping te-node-tunnel-termination-point-config-attributes {  
    description  
      "Configuration attributes of a tunnel termination point on a  
      TE node.";  
    leaf admin-status {  
      type te-types:te-admin-status;  
      description  
        "The administrative state of the tunnel termination point.";  
    }  
    leaf name {  
      type string;  
      description  
        "A descriptive name for the tunnel termination point.";  
    }  
  }
```

```
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.";
}
leaf-list inter-layer-lock-id {
  type uint32;
  description
    "Inter layer lock ID, used for path computation in a TE
    topology covering multiple layers or multiple regions.";
  reference
    "RFC 5212: Requirements for GMPLS-Based Multi-Region and
    Multi-Layer Networks (MRN/MLN).
    RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
    for Multi-Layer and Multi-Region Networks (MLN/MRN).";
}
leaf protection-type {
  type identityref {
    base te-types:lsp-protection-type;
  }
  description
    "The protection type that this tunnel termination point
    is capable of.";
}

container client-layer-adaptation {
  description
    "Containing capability information to support a client layer
    adaption in multi-layer topology.";
  list switching-capability {
    key "switching-capability encoding";
    description
```

```
        "List of supported switching capabilities";
    reference
        "RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
        for Multi-Layer and Multi-Region Networks (MLN/MRN).
        RFC 4202: Routing Extensions in Support of
        Generalized Multi-Protocol Label Switching (GMPLS).";
    leaf switching-capability {
        type identityref {
            base te-types:switching-capabilities;
        }
        description
            "Switching Capability for the client layer adaption.";
    }
    leaf encoding {
        type identityref {
            base te-types:lsp-encoding-types;
        }
        description
            "Encoding supported by the client layer adaption.";
    }
    uses te-types:te-bandwidth;
}
}
} // te-node-tunnel-termination-point-config-attributes

grouping te-node-tunnel-termination-point-llc-list {
    description
        "Local link connectivity list of a tunnel termination
        point on a TE node.";
    list local-link-connectivity {
        key "link-tp-ref";
        description
            "The termination capabilities between
            tunnel-termination-point and link termination-point.
            The capability information can be used to compute
            the tunnel path.
            The Interface Adjustment Capability Descriptors (IACD)
            (defined in RFC 6001) on each link-tp can be derived from
            this local-link-connectivity list.";
        reference
            "RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
```

```
        for Multi-Layer and Multi-Region Networks (MLN/MRN).";

    leaf link-tp-ref {
        type leafref {
            path "../..../..../nt:termination-point/nt:tp-id";
        }
        description
            "Link termination point.";
    }
    uses te-types:label-set-info;
    uses connectivity-matrix-entry-path-attributes;
} // local-link-connectivity
} // te-node-tunnel-termination-point-config

grouping te-path-element {
    description
        "A group of attributes defining an element in a TE path
        such as TE node, TE link, TE atomic resource or label.";
    uses te-types:explicit-route-hop;
} // te-path-element

grouping te-termination-point-augment {
    description
        "Augmentation for TE termination point.";
    leaf te-tp-id {
        type te-types:te-tp-id;
        description
            "An identifier to uniquely identify a TE termination
            point.";
    }
    container te {
        must "../te-tp-id";
        presence "TE support.";
        description
            "Indicates TE support.";

        uses te-termination-point-config;
        leaf oper-status {
            type te-types:te-oper-status;
            config false;
            description

```



```
        "The current operational state of the link termination
        point.";
    }
    uses geolocation-container;
} // te
} // te-termination-point-augment

grouping te-termination-point-config {
    description
        "TE termination point configuration grouping.";
    leaf admin-status {
        type te-types:te-admin-status;
        description
            "The administrative state of the link termination point.";
    }
    leaf name {
        type string;
        description
            "A descriptive name for the link termination point.";
    }
    uses interface-switching-capability-list;
    leaf inter-domain-plug-id {
        type binary;
        description
            "A topology-wide unique number that identifies on the
            network a connectivity supporting a given inter-domain
            TE link. This is more flexible alternative to specifying
            remote-te-node-id and remote-te-link-tp-id on a TE link,
            when the provider does not know remote-te-node-id and
            remote-te-link-tp-id or need to give client the
            flexibility to mix-n-match multiple topologies.";
    }
    leaf-list inter-layer-lock-id {
        type uint32;
        description
            "Inter layer lock ID, used for path computation in a TE
            topology covering multiple layers or multiple regions.";
        reference
            "RFC 5212: Requirements for GMPLS-Based Multi-Region and
            Multi-Layer Networks (MRN/MLN).
            RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
```

```
        for Multi-Layer and Multi-Region Networks (MLN/MRN).";
    }
} // te-termination-point-config

grouping te-topologies-augment {
  description
    "Augmentation for TE topologies.";
  container te {
    presence "TE support.";
    description
      "Indicates TE support.";

    container templates {
      description
        "Configuration parameters for templates used for TE
        topology.";

      list node-template {
        if-feature template;
        key "name";
        leaf name {
          type te-types:te-template-name;
          description
            "The name to identify a TE node template.";
        }
        description
          "The list of TE node templates used to define sharable
          and reusable TE node attributes.";
        uses template-attributes;
        uses te-node-config-attributes-template;
      } // node-template

      list link-template {
        if-feature template;
        key "name";
        leaf name {
          type te-types:te-template-name;
          description
            "The name to identify a TE link template.";
        }
        description

```

```
        "The list of TE link templates used to define sharable
        and reusable TE link attributes.";
    uses template-attributes;
    uses te-link-config-attributes;
    } // link-template
} // templates
} // te
} // te-topologies-augment

grouping te-topology-augment {
    description
        "Augmentation for TE topology.";
    uses te-types:te-topology-identifier;

    container te {
        must "../te-topology-identifier/provider-id"
            + " and ../te-topology-identifier/client-id"
            + " and ../te-topology-identifier/topology-id";
        presence "TE support.";
        description
            "Indicates TE support.";

        uses te-topology-config;
        uses geolocation-container;
    } // te
} // te-topology-augment

grouping te-topology-config {
    description
        "TE topology configuration grouping.";
    leaf name {
        type string;
        description
            "Name of the TE topology. This attribute is optional and can
            be specified by the operator to describe the TE topology,
            which can be useful when network-id is not descriptive
            and not modifiable because of being generated by the
            system.";
    }
    leaf preference {
        type uint8 {
```

```
        range "1..255";
    }
    description
        "Specifies a preference for this topology. A lower number
        indicates a higher preference.";
}
leaf optimization-criterion {
    type identityref {
        base te-types:objective-function-type;
    }
    description
        "Optimization criterion applied to this topology.";
    reference
        "RFC 3272: Overview and Principles of Internet Traffic
        Engineering.";
}
list nsrlg {
    if-feature nsrlg;
    key "id";
    description
        "List of NSRLGs (Not Sharing Risk Link Groups).";
    reference
        "RFC 4872: RSVP-TE Extensions in Support of End-to-End
        Generalized Multi-Protocol Label Switching (GMPLS)
        Recovery";
    leaf id {
        type uint32;
        description
            "Identify the NSRLG entry.";
    }
    leaf disjointness {
        type te-types:te-path-disjointness;
        description
            "The type of resource disjointness.";
    }
} // nsrlg
} // te-topology-config

grouping template-attributes {
    description
        "Common attributes for all templates.";
```

```
leaf priority {
  type uint16;
  description
    "The preference value to resolve conflicts between different
    templates. When two or more templates specify values for
    one configuration attribute, the value from the template
    with the highest priority is used.
    A lower number indicates a higher priority. The highest
    priority is 0.";
}
leaf reference-change-policy {
  type enumeration {
    enum no-action {
      description
        "When an attribute changes in this template, the
        configuration node referring to this template does
        not take any action.";
    }
    enum not-allowed {
      description
        "When any configuration object has a reference to this
        template, changing this template is not allowed.";
    }
    enum cascade {
      description
        "When an attribute changes in this template, the
        configuration object referring to this template applies
        the new attribute value to the corresponding
        configuration.";
    }
  }
  description
    "This attribute specifies the action taken to a configuration
    node that has a reference to this template.";
}
} // template-attributes

/*
 * Data nodes
 */
augment "/nw:networks/nw:network/nw:network-types" {
```

```
    description
      "Introduce new network type for TE topology.";
    container te-topology {
      presence "Indicates TE topology.";
      description
        "Its presence identifies the TE topology type.";
    }
  }

  augment "/nw:networks" {
    description
      "Augmentation parameters for TE topologies.";
    uses te-topologies-augment;
  }

  augment "/nw:networks/nw:network" {
    when "nw:network-types/tet:te-topology" {
      description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
      "Configuration parameters for TE topology.";
    uses te-topology-augment;
  }

  augment "/nw:networks/nw:network/nw:node" {
    when "../nw:network-types/tet:te-topology" {
      description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
      "Configuration parameters for TE at node level.";
    leaf te-node-id {
      type te-types:te-node-id;
      description
        "The identifier of a node in the TE topology.
        A node is specific to a topology to which it belongs.";
    }
  }
  container te {
```

```
    must "../te-node-id" {
      description
        "te-node-id is mandatory.";
    }
    must "count(..../nw:supporting-node)<=1" {
      description
        "For a node in a TE topology, there cannot be more
        than 1 supporting node. If multiple nodes are abstracted,
        the underlay-topology is used.";
    }
    presence "TE support.";
    description
      "Indicates TE support.";
    uses te-node-augment;
  } // te
}

augment "/nw:networks/nw:network/nt:link" {
  when "../nw:network-types/tet:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Configuration parameters for TE at link level.";
  container te {
    must "count(..../nt:supporting-link)<=1" {
      description
        "For a link in a TE topology, there cannot be more
        than 1 supporting link. If one or more link paths are
        abstracted, the underlay is used.";
    }
    presence "TE support.";
    description
      "Indicates TE support.";
    uses te-link-augment;
  } // te
}

augment "/nw:networks/nw:network/nw:node/"
  + "nt:termination-point" {
```

```
when "../../../nw:network-types/tet:te-topology" {
  description
    "Augmentation parameters apply only for networks with
    TE topology type.";
}
description
  "Configuration parameters for TE at termination point level.";
uses te-termination-point-augment;
}

augment
  "/nw:networks/nw:network/nt:link/te/bundle-stack-level/"
+ "bundle/bundled-links/bundled-link" {
  when "../../../nw:network-types/tet:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE link bundled link.";
  leaf src-tp-ref {
    type leafref {
      path "../../../nw:node[nw:node-id = "
        + "current()/../../../../nt:source/"
        + "nt:source-node]/"
        + "nt:termination-point/nt:tp-id";
      require-instance true;
    }
    description
      "Reference to another TE termination point on the
      same source node.";
  }
  leaf des-tp-ref {
    type leafref {
      path "../../../nw:node[nw:node-id = "
        + "current()/../../../../nt:destination/"
        + "nt:dest-node]/"
        + "nt:termination-point/nt:tp-id";
      require-instance true;
    }
    description

```



```
        "Reference to another TE termination point on the
        same destination node.";
    }
}

augment
  "/nw:networks/nw:network/nw:node/te/"
+ "information-source-entry/connectivity-matrices/"
+ "connectivity-matrix" {
  when "../..../nw:network-types/tet:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
  "/nw:networks/nw:network/nw:node/te/te-node-attributes/"
+ "connectivity-matrices/connectivity-matrix" {
  when "../..../nw:network-types/tet:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
  "/nw:networks/nw:network/nw:node/te/"
+ "tunnel-termination-point/local-link-connectivities" {
  when "../..../nw:network-types/tet:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
```

```
    "Augment TE node tunnel termination point LLCs
      (Local Link Connectivities).";
    uses te-node-tunnel-termination-point-llc-list;
  }
}
<CODE ENDS>
```

## 8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The 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:

- o /nw:networks/nw:network/nw:network-types/tet:te-topology  
This subtree specifies the TE topology type. Modifying the configurations can make TE topology type invalid. By such modifications, a malicious attacker may disable the TE capabilities on the related networks and cause traffic disrupted or misrouted.
- o /nw:networks/tet:te  
This subtree specifies the TE node templates and TE link templates. Modifying the configurations in this subtree will change the related future TE configurations. By such modifications, a malicious attacker may change the TE capabilities scheduled at a future time, to cause traffic disrupted or misrouted.

- o /nw:networks/nw:network  
This subtree specifies the topology-wide configurations, including the TE topology ID and topology-wide policies. Modifying the configurations in this subtree can add, remove, or modify TE topologies. By adding a TE topology, a malicious attacker may create an unauthorized traffic network. By removing or modifying a TE topology, a malicious attacker may cause traffic disabled or misrouted in the specified TE topology. Such traffic changes may also affect the traffic in the connected TE topologies.
- o /nw:networks/nw:network/nw:node  
This subtree specifies the configurations for TE nodes. Modifying the configurations in this subtree can add, remove, or modify TE nodes. By adding a TE node, a malicious attacker may create an unauthorized traffic path. By removing or modifying a TE node, a malicious attacker may cause traffic disabled or misrouted in the specified TE node. Such traffic changes may also affect the traffic on the surrounding TE nodes and TE links in this TE topology and the connected TE topologies.
- o /nw:networks/nw:network/nt:link/tet:te  
This subtree specifies the configurations for TE links. Modifying the configurations in this subtree can add, remove, or modify TE links. By adding a TE link, a malicious attacker may create an unauthorized traffic path. By removing or modifying a TE link, a malicious attacker may cause traffic disabled or misrouted on the specified TE link. Such traffic changes may also affect the traffic on the surrounding TE nodes and TE links in this TE topology and the connected TE topologies.
- o /nw:networks/nw:network/nw:node/nt:termination-point  
This subtree specifies the configurations of TE link termination points. Modifying the configurations in this subtree can add, remove, or modify TE link termination points. By adding a TE link termination point, a malicious attacker may create an unauthorized traffic path. By removing or modifying a TE link termination point, a malicious attacker may cause traffic disabled or misrouted on the specified TE link termination point. Such traffic changes may also affect the traffic on the surrounding TE nodes and TE links in this TE topology and the connected TE topologies.

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

- o /nw:networks/nw:network/nw:network-types/tet:te-topology  
Unauthorized access to this subtree can disclose the TE topology type.
- o /nw:networks/tet:te  
Unauthorized access to this subtree can disclose the TE node templates and TE link templates.
- o /nw:networks/nw:network  
Unauthorized access to this subtree can disclose the topology-wide configurations, including the TE topology ID, the topology-wide policies, and the topology geolocation.
- o /nw:networks/nw:network/nw:node  
Unauthorized access to this subtree can disclose the operational state information of TE nodes.
- o /nw:networks/nw:network/nt:link/tet:te  
Unauthorized access to this subtree can disclose the operational state information of TE links.
- o /nw:networks/nw:network/nw:node/nt:termination-point  
Unauthorized access to this subtree can disclose the operational state information of TE link termination points.

## 9. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

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

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

This document registers a YANG module in the YANG Module Names registry [RFC7950].

name: ietf-te-topology  
namespace: urn:ietf:params:xml:ns:yang:ietf-te-topology  
prefix: tet  
reference: RFC XXXX

name: ietf-te-topology-state  
namespace: urn:ietf:params:xml:ns:yang:ietf-te-topology-state  
prefix: tet-s  
reference: RFC XXXX

## 10. References

### 10.1. 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>>.
- [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>>.
- [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>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7926] Farrel, A., Ed., Drake, J., Bitar, N., Swallow, G., Ceccarelli, D., and X. Zhang, "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", BCP 206, RFC 7926, DOI 10.17487/RFC7926, July 2016, <<https://www.rfc-editor.org/info/rfc7926>>.
- [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>>.
- [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>>.
- [RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <<https://www.rfc-editor.org/info/rfc8345>>.
- [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>>.
- [I-D.ietf-teas-yang-te-types]  
Saad, T., Gandhi, R., Liu, X., Beeram, V., and  
I. Bryskin, "Traffic Engineering Common YANG Types",  
draft-ietf-teas-yang-te-types-08 (work in progress),  
April 2019.

## 10.2. Informative References

- [G.709] ITU-T, "Interfaces for the optical transport network", ITU-T Recommendation G.709, June 2016.
- [G.805] ITU-T, "Generic functional architecture of transport networks", ITU-T Recommendation G.805, March 2000.
- [G.872] ITU-T, "Architecture of optical transport networks", ITU-T Recommendation G.872, January 2017.
- [G.8080] ITU-T, "Architecture for the automatically switched optical network", ITU-T Recommendation G.8080, February 2012.

- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [RFC2702] Awduche, D., Malcolm, J., Agogbua, J., O'Dell, M., and J. McManus, "Requirements for Traffic Engineering Over MPLS", RFC 2702, DOI 10.17487/RFC2702, September 1999, <<https://www.rfc-editor.org/info/rfc2702>>.
- [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>>.
- [RFC3272] Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., and X. Xiao, "Overview and Principles of Internet Traffic Engineering", RFC 3272, DOI 10.17487/RFC3272, May 2002, <<https://www.rfc-editor.org/info/rfc3272>>.
- [RFC3471] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, DOI 10.17487/RFC3471, January 2003, <<https://www.rfc-editor.org/info/rfc3471>>.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <<https://www.rfc-editor.org/info/rfc3630>>.
- [RFC3785] Le Faucheur, F., Uppili, R., Vedrenne, A., Merckx, P., and T. Telkamp, "Use of Interior Gateway Protocol (IGP) Metric as a second MPLS Traffic Engineering (TE) Metric", BCP 87, RFC 3785, DOI 10.17487/RFC3785, May 2004, <<https://www.rfc-editor.org/info/rfc3785>>.
- [RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, DOI 10.17487/RFC4201, October 2005, <<https://www.rfc-editor.org/info/rfc4201>>.
- [RFC4202] Kompella, K., Ed. and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, DOI 10.17487/RFC4202, October 2005, <<https://www.rfc-editor.org/info/rfc4202>>.
- [RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching

- (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <<https://www.rfc-editor.org/info/rfc4203>>.
- [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., 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>>.
- [RFC5152] Vasseur, JP., Ed., Ayyangar, A., Ed., and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)", RFC 5152, DOI 10.17487/RFC5152, February 2008, <<https://www.rfc-editor.org/info/rfc5152>>.
- [RFC5212] Shiimoto, K., Papadimitriou, D., Le Roux, JL., Vigoureux, M., and D. Brungard, "Requirements for GMPLS-Based Multi-Region and Multi-Layer Networks (MRN/MLN)", RFC 5212, DOI 10.17487/RFC5212, July 2008, <<https://www.rfc-editor.org/info/rfc5212>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, DOI 10.17487/RFC5316, December 2008, <<https://www.rfc-editor.org/info/rfc5316>>.
- [RFC5329] Ishiguro, K., Manral, V., Davey, A., and A. Lindem, Ed., "Traffic Engineering Extensions to OSPF Version 3", RFC 5329, DOI 10.17487/RFC5329, September 2008, <<https://www.rfc-editor.org/info/rfc5329>>.
- [RFC5392] Chen, M., Zhang, R., and X. Duan, "OSPF Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5392, DOI 10.17487/RFC5392, January 2009, <<https://www.rfc-editor.org/info/rfc5392>>.
- [RFC6001] Papadimitriou, D., Vigoureux, M., Shiimoto, K., Brungard,



- D., and JL. Le Roux, "Generalized MPLS (GMPLS) Protocol Extensions for Multi-Layer and Multi-Region Networks (MLN/MRN)", RFC 6001, DOI 10.17487/RFC6001, October 2010, <<https://www.rfc-editor.org/info/rfc6001>>.
- [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>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <<https://www.rfc-editor.org/info/rfc7471>>.
- [RFC7579] Bernstein, G., Ed., Lee, Y., Ed., Li, D., Imajuku, W., and J. Han, "General Network Element Constraint Encoding for GMPLS-Controlled Networks", RFC 7579, DOI 10.17487/RFC7579, June 2015, <<https://www.rfc-editor.org/info/rfc7579>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.
- [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>>.
- [I-D.ietf-netconf-subscribed-notifications]  
Voit, E., Clemm, A., Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Customized Subscriptions to a Publisher's Event Streams", draft-ietf-netconf-subscribed-notifications-23 (work in progress), February 2019.
- [I-D.ietf-netconf-yang-push]  
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", draft-ietf-netconf-yang-push-22 (work in progress), February 2019.
- [I-D.liu-netmod-yang-schedule]  
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "A YANG Data Model for Configuration Scheduling", draft-liu-netmod-yang-schedule-05 (work in progress),

March 2018.

[I-D.ietf-ccamp-wson-yang]

Lee, Y., Dhody, D., Zhang, X., Guo, A., Lopezalvarez, V., King, D., Yoon, B., and R. Vilata, "A Yang Data Model for WSON Optical Networks", draft-ietf-ccamp-wson-yang-20 (work in progress), March 2019.

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

zhenghaomian@huawei.com, z., Guo, A., Busi, I., Sharma, A., Liu, X., Belotti, S., Xu, Y., Wang, L., and O. Dios, "A YANG Data Model for Optical Transport Network Topology", draft-ietf-ccamp-otn-topo-yang-06 (work in progress), February 2019.

[I-D.ietf-teas-yang-l3-te-topo]

Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Layer 3 TE Topologies", draft-ietf-teas-yang-l3-te-topo-04 (work in progress), March 2019.

[I-D.ietf-teas-te-topo-and-tunnel-modeling]

Bryskin, I., Beeram, V., Saad, T., and X. Liu, "TE Topology and Tunnel Modeling for Transport Networks", draft-ietf-teas-te-topo-and-tunnel-modeling-03 (work in progress), October 2018.

## 11. Acknowledgments

The authors would like to thank Lou Berger, Sue Hares, Mazen Khaddam, Cyril Margaria and Zafar Ali for participating in design discussions and providing valuable insights.

## Appendix A. Complete Model Tree Structure

```

module: ietf-te-topology
  augment /nw:networks/nw:network/nw:network-types:
    +--rw te-topology!
  augment /nw:networks:
    +--rw te!
      +--rw templates
        +--rw node-template* [name] {template}?
          +--rw name
            | te-types:te-template-name
          +--rw priority? uint16
          +--rw reference-change-policy? enumeration
          +--rw te-node-attributes
            +--rw admin-status? te-types:te-admin-status
            +--rw domain-id? uint32
            +--rw is-abstract? empty
            +--rw name? string
            +--rw signaling-address* inet:ip-address
            +--rw underlay-topology {te-topology-hierarchy}?
              +--rw network-ref?
                -> /nw:networks/network/network-id
        +--rw link-template* [name] {template}?
          +--rw name
            | te-types:te-template-name
          +--rw priority? uint16
          +--rw reference-change-policy? enumeration
          +--rw te-link-attributes
            +--rw access-type?
              | te-types:te-link-access-type
            +--rw external-domain
              +--rw network-ref?
                | -> /nw:networks/network/network-id
                +--rw remote-te-node-id? te-types:te-node-id
                +--rw remote-te-link-tp-id? te-types:te-tp-id
            +--rw is-abstract? empty
            +--rw name? string
            +--rw underlay {te-topology-hierarchy}?
              +--rw enabled? boolean
              +--rw primary-path
                | +--rw network-ref?

```

```

|         -> /nw:networks/network/network-id
+--rw path-element* [path-element-id]
|   +--rw path-element-id          uint32
|   +--rw (type)?
|     +--:(numbered-node-hop)
|       +--rw numbered-node-hop
|         +--rw node-id          te-node-id
|         +--rw hop-type?       te-hop-type
|     +--:(numbered-link-hop)
|       +--rw numbered-link-hop
|         +--rw link-tp-id       te-tp-id
|         +--rw hop-type?       te-hop-type
|         +--rw direction?
|           te-link-direction
|     +--:(unnumbered-link-hop)
|       +--rw unnumbered-link-hop
|         +--rw link-tp-id       te-tp-id
|         +--rw node-id          te-node-id
|         +--rw hop-type?       te-hop-type
|         +--rw direction?
|           te-link-direction
|     +--:(as-number)
|       +--rw as-number-hop
|         +--rw as-number        inet:as-number
|         +--rw hop-type?       te-hop-type
|     +--:(label)
|       +--rw label-hop
|         +--rw te-label
|           +--rw (technology)?
|             +--:(generic)
|               +--rw generic?
|                 rt-
types:generalized-label
|         +--rw direction?
|           te-label-direction
+--rw backup-path* [index]
|   +--rw index          uint32
|   +--rw network-ref?
|     -> /nw:networks/network/network-id
+--rw path-element* [path-element-id]
|   +--rw path-element-id          uint32

```

```

+--rw (type)?
+--:(numbered-node-hop)
|   +--rw numbered-node-hop
|       +--rw node-id      te-node-id
|       +--rw hop-type?    te-hop-type
+--:(numbered-link-hop)
|   +--rw numbered-link-hop
|       +--rw link-tp-id    te-tp-id
|       +--rw hop-type?    te-hop-type
|       +--rw direction?
|           te-link-direction
+--:(unnumbered-link-hop)
|   +--rw unnumbered-link-hop
|       +--rw link-tp-id    te-tp-id
|       +--rw node-id      te-node-id
|       +--rw hop-type?    te-hop-type
|       +--rw direction?
|           te-link-direction
+--:(as-number)
|   +--rw as-number-hop
|       +--rw as-number     inet:as-number
|       +--rw hop-type?    te-hop-type
+--:(label)
|   +--rw label-hop
|       +--rw te-label
|           +--rw (technology)?
|               +--:(generic)
|                   +--rw generic?
|                       rt-
types:generalized-label
|       +--rw direction?
|           te-label-direction
+--rw protection-type?          identityref
+--rw tunnel-termination-points
|   +--rw source?                binary
|   +--rw destination?          binary
+--rw tunnels
|   +--rw sharing?              boolean
|   +--rw tunnel* [tunnel-name]
|       +--rw tunnel-name        string
|       +--rw sharing?            boolean

```

```

+--rw admin-status?
|   te-types:te-admin-status
+--rw link-index?                               uint64
+--rw administrative-group?
|   te-types:admin-groups
+--rw interface-switching-capability*
|   [switching-capability encoding]
|   +--rw switching-capability identityref
|   +--rw encoding identityref
|   +--rw max-lsp-bandwidth* [priority]
|       +--rw priority uint8
|       +--rw te-bandwidth
|           +--rw (technology)?
|               +--:(generic)
|                   +--rw generic? te-bandwidth
+--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 link-protection-type? identityref
+--rw max-link-bandwidth
|   +--rw te-bandwidth

```

```

    |         +---rw (technology)?
    |         |         +---:(generic)
    |         |         +---rw generic?    te-bandwidth
+---rw max-resv-link-bandwidth
    |         +---rw te-bandwidth
    |         |         +---rw (technology)?
    |         |         +---:(generic)
    |         |         +---rw generic?    te-bandwidth
+---rw unreserved-bandwidth* [priority]
    |         +---rw priority            uint8
    |         +---rw te-bandwidth
    |         |         +---rw (technology)?
    |         |         +---:(generic)
    |         |         +---rw generic?    te-bandwidth
+---rw te-default-metric?                uint32
+---rw te-delay-metric?                  uint32
+---rw te-igp-metric?                    uint32
+---rw te-srlgs
    |   +---rw value*    te-types:srlg
+---rw te-nsrlgs {nsrlg}?
    |   +---rw id*      uint32
augment /nw:networks/nw:network:
+---rw te-topology-identifier
    |   +---rw provider-id?    te-global-id
    |   +---rw client-id?      te-global-id
    |   +---rw topology-id?    te-topology-id
+---rw te!
    |   +---rw name?            string
    |   +---rw preference?      uint8
    |   +---rw optimization-criterion?  identityref
    |   +---rw nsrlg* [id] {nsrlg}?
    |   |   +---rw id          uint32
    |   |   +---rw disjointness? te-types:te-path-disjointness
+---ro geolocation
    |   +---ro altitude?        int64
    |   +---ro latitude?        geographic-coordinate-degree
    |   +---ro longitude?       geographic-coordinate-degree
augment /nw:networks/nw:network/nw:node:
+---rw te-node-id?    te-types:te-node-id
+---rw te!
    |   +---rw te-node-template*

```

```

|         -> ../../../../te/templates/node-template/name
|         {template}?
+--rw te-node-attributes
|   +--rw admin-status?          te-types:te-admin-status
|   +--rw connectivity-matrices
|   |   +--rw number-of-entries?    uint16
|   |   +--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 is-allowed?              boolean
|   +--rw underlay {te-topology-hierarchy}?
|   |   +--rw enabled?                  boolean
|   |   +--rw primary-path
|   |   |   +--rw network-ref?
|   |   |   |   -> /nw:networks/network/network-id
|   |   |   +--rw path-element* [path-element-id]
|   |   |   |   +--rw path-element-id          uint32
|   |   |   |   +--rw (type)?
|   |   |   |   |   +--:(numbered-node-hop)
|   |   |   |   |   |   +--rw numbered-node-hop
|   |   |   |   |   |   |   +--rw node-id          te-node-id

```





|       |  |  |  |  |                                 |                   |
|-------|--|--|--|--|---------------------------------|-------------------|
|       |  |  |  |  | +--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 protection-type?          | identityref       |
|       |  |  |  |  | +--rw tunnel-termination-points |                   |
|       |  |  |  |  | +--rw source?                   | binary            |
|       |  |  |  |  | +--rw destination?              | binary            |
|       |  |  |  |  | +--rw tunnels                   |                   |
|       |  |  |  |  | +--rw sharing?                  | boolean           |
|       |  |  |  |  | +--rw tunnel*                   | [tunnel-name]     |
|       |  |  |  |  | +--rw tunnel-name               | string            |
|       |  |  |  |  | +--rw sharing?                  | boolean           |
|       |  |  |  |  | +--rw path-constraints          |                   |
|       |  |  |  |  | +--rw te-bandwidth              |                   |
|       |  |  |  |  | +--rw (technology)?             |                   |
|       |  |  |  |  | +---:(generic)                  |                   |
|       |  |  |  |  | +--rw generic?                  | te-bandwidth      |
|       |  |  |  |  | +--rw link-protection?          | identityref       |
|       |  |  |  |  | +--rw setup-priority?           | uint8             |
|       |  |  |  |  | +--rw hold-priority?            | uint8             |
|       |  |  |  |  | +--rw signaling-type?           | identityref       |
|       |  |  |  |  | +--rw path-metric-bounds        |                   |
|       |  |  |  |  | +--rw path-metric-bound*        | [metric-type]     |

```

|         +---rw metric-type      identityref
|         +---rw upper-bound?    uint64
+---rw path-affinities-values
|         +---rw path-affinities-value* [usage]
|         +---rw usage            identityref
|         +---rw value?          admin-groups
+---rw path-affinity-names
|         +---rw path-affinity-name* [usage]
|         +---rw usage            identityref
|         +---rw affinity-name* [name]
|         +---rw name             string
+---rw path-srlgs-lists
|         +---rw path-srlgs-list* [usage]
|         +---rw usage            identityref
|         +---rw values*          srlg
+---rw path-srlgs-names
|         +---rw path-srlgs-name* [usage]
|         +---rw usage            identityref
|         +---rw names*           string
+---rw disjointness?              te-path-disjointness
+---rw optimizations
+---rw (algorithm)?
|         +---:(metric) {path-optimization-metric}?
|         |         +---rw optimization-metric* [metric-type]
|         |         |         +---rw metric-type
|         |         |         |         identityref
|         |         |         +---rw weight?
|         |         |         |         uint8
|         |         |         +---rw explicit-route-exclude-objects
|         |         |         |         +---rw route-object-exclude-object*
|         |         |         |         |         [index]
|         |         |         |         +---rw index
|         |         |         |         |         uint32
|         |         |         |         +---rw (type)?
|         |         |         |         |         +---:(numbered-node-hop)
|         |         |         |         |         |         +---rw numbered-node-hop
|         |         |         |         |         |         |         +---rw node-id      te-node-id
|         |         |         |         |         |         |         +---rw hop-type?   te-hop-type
|         |         |         |         |         +---:(numbered-link-hop)
|         |         |         |         |         |         +---rw numbered-link-hop
|         |         |         |         |         |         +---rw link-tp-id      te-tp-id

```

```

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

```

```

+---rw hop-type?      te-hop-type
+---:(numbered-link-hop)
|   +---rw numbered-link-hop
|       +---rw link-tp-id      te-tp-id
|       +---rw hop-type?
|           |
|           te-hop-type
|       +---rw direction?
|           te-link-direction
+---:(unnumbered-link-hop)
|   +---rw unnumbered-link-hop
|       +---rw link-tp-id      te-tp-id
|       +---rw node-id
|           |
|           te-node-id
|       +---rw hop-type?
|           |
|           te-hop-type
|       +---rw direction?
|           te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|       +---rw as-number
|           |
|           inet:as-number
|       +---rw hop-type?
|           te-hop-type
+---:(label)
|   +---rw label-hop
|       +---rw te-label
|           +---rw (technology)?
|               +---:(generic)
|                   +---rw generic?
|                       rt-
types:generalized-label
|
|       +---rw direction?
|           te-label-direction
+---rw tiebreakers
+---rw tiebreaker* [tiebreaker-type]
+---rw tiebreaker-type      identityref
+---:(objective-function)
|   {path-optimization-objective-function}?
+---rw objective-function
+---rw objective-function-type?      identityref
+---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:generalized- |                |
| label |  |                            |                     |                       |                |
|       |  |                            |                     | +--ro direction?      |                |
|       |  |                            |                     | te-label-direction    |                |
|       |  | +--rw connectivity-matrix* | [id]                |                       |                |
|       |  | +--rw id                   |                     | uint32                |                |
|       |  | +--rw from                 |                     |                       |                |
|       |  | +--rw tp-ref?              |                     | leafref               |                |
|       |  | +--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 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?
+---rw 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?
+---rw 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?
+---rw rt-
types:generalized-label
+---rw direction?
+---rw te-label-direction
+---rw backup-path* [index]
+---rw index            uint32
+---rw network-ref?
+---rw -> /nw:networks/network/network-id
+---rw path-element* [path-element-id]
+---rw path-element-id  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?
+---rw 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?
+---rw 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?
+---rw rt-
types:generalized-label
+---rw direction?
+---rw te-label-direction
+---rw protection-type?      identityref
+---rw tunnel-termination-points
+---rw source?               binary
+---rw destination?         binary
+---rw tunnels
+---rw sharing?              boolean
+---rw tunnel* [tunnel-name]
+---rw tunnel-name           string
+---rw sharing?              boolean
+---rw path-constraints
+---rw te-bandwidth
+---rw (technology)?
+---:(generic)
+---rw generic?              te-bandwidth

```

```

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

```

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

```

+--:(srlg)
  +--rw srlg
    +--rw srlg? uint32
+--rw explicit-route-include-objects
  +--rw route-object-include-object*
    [index]
  +--rw index
    | uint32
  +--rw (type)?
    +--:(numbered-node-hop)
      +--rw numbered-node-hop
        +--rw node-id
          | te-node-id
        +--rw hop-type?
          | te-hop-type
    +--:(numbered-link-hop)
      +--rw numbered-link-hop
        +--rw link-tp-id
          | te-tp-id
        +--rw hop-type?
          | te-hop-type
        +--rw direction?
          | te-link-direction
    +--:(unnumbered-link-hop)
      +--rw unnumbered-link-hop
        +--rw link-tp-id
          | te-tp-id
        +--rw node-id
          | te-node-id
        +--rw hop-type?
          | te-hop-type
        +--rw direction?
          | te-link-direction
    +--:(as-number)
      +--rw as-number-hop
        +--rw as-number
          | inet:as-number
        +--rw hop-type?
          | te-hop-type
    +--:(label)
      +--rw label-hop

```

```

+---rw te-label
+---rw (technology)?
+---:(generic)
+---rw generic?
rt-

types:generalized-label

+---rw direction?
te-label-

direction

+---rw tiebreakers
+---rw tiebreaker* [tiebreaker-type]
+---rw tiebreaker-type identityref
+---:(objective-function)
{path-optimization-objective-

function}?

+---rw objective-function
+---rw objective-function-type?
identityref

+---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-
          +--ro direction?
            te-label-direction
types:generalized-label
+--rw domain-id?                uint32
+--rw is-abstract?              empty
+--rw name?                     string
+--rw signaling-address*        inet:ip-address
+--rw underlay-topology {te-topology-hierarchy}?
  +--rw network-ref?    -> /nw:networks/network/network-id
+--ro oper-status?            te-types:te-oper-status
+--ro geolocation

```

```

    |   +---ro altitude?      int64
    |   +---ro latitude?     geographic-coordinate-degree
    |   +---ro longitude?    geographic-coordinate-degree
+---ro is-multi-access-dr?      empty
+---ro information-source?      te-info-source
+---ro information-source-instance? string
+---ro information-source-state
    |   +---ro credibility-preference? uint16
    |   +---ro logical-network-element? string
    |   +---ro network-instance?      string
    |   +---ro topology
    |       +---ro node-ref?      leafref
    |       +---ro network-ref?  -> /nw:networks/network/network-id
+---ro information-source-entry*
    |   [information-source information-source-instance]
    |   +---ro information-source      te-info-source
    |   +---ro information-source-instance string
    |   +---ro information-source-state
    |       +---ro credibility-preference? uint16
    |       +---ro logical-network-element? string
    |       +---ro network-instance?      string
    |       +---ro topology
    |           +---ro node-ref?      leafref
    |           +---ro network-ref?
    |               -> /nw:networks/network/network-id
+---ro connectivity-matrices
    |   +---ro number-of-entries?      uint16
    |   +---ro label-restrictions
    |       +---ro label-restriction* [index]
    |           +---ro restriction?      enumeration
    |           +---ro index            uint32
    |           +---ro label-start
    |               +---ro te-label
    |                   +---ro (technology)?
    |                       +---: (generic)
    |                           +---ro generic?
    |                               rt-types:generalized-label
    |               +---ro direction?      te-label-direction
    |   +---ro label-end
    |       +---ro te-label
    |           +---ro (technology)?

```



```

+---:(generic)
+---ro generic?
+---ro rt-types:generalized-label
+---ro direction?      te-label-direction
+---ro label-step
+---ro (technology)?
+---:(generic)
+---ro generic?      int32
+---ro range-bitmap?  yang:hex-string
+---ro is-allowed?    boolean
+---ro underlay {te-topology-hierarchy}?
+---ro enabled?      boolean
+---ro primary-path
+---ro network-ref?
+---ro -> /nw:networks/network/network-id
+---ro path-element* [path-element-id]
+---ro path-element-id      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)?

```

|       |  |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------|--|--|--|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  |  |  | <pre> +--:(generic)   +--ro generic?     rt-types:generalized- </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| label |  |  |  |  | <pre> +--ro direction?   te-label-direction +--ro backup-path* [index]   +--ro index          uint32   +--ro network-ref?             -&gt; /nw:networks/network/network-id   +--ro path-element* [path-element-id]     +--ro path-element-id          uint32     +--ro (type)?       +--:(numbered-node-hop)         +--ro numbered-node-hop           +--ro node-id      te-node-id           +--ro hop-type?    te-hop-type       +--:(numbered-link-hop)         +--ro numbered-link-hop           +--ro link-tp-id    te-tp-id           +--ro hop-type?    te-hop-type           +--ro direction?   te-link-direction       +--:(unnumbered-link-hop)         +--ro unnumbered-link-hop           +--ro link-tp-id    te-tp-id           +--ro node-id      te-node-id           +--ro hop-type?    te-hop-type           +--ro direction?   te-link-direction       +--:(as-number)         +--ro as-number-hop           +--ro as-number    inet:as-number           +--ro hop-type?    te-hop-type       +--:(label)         +--ro label-hop           +--ro te-label             +--ro (technology)?               +--:(generic)                 +--ro generic?                   rt-types:generalized- </pre> |
| label |  |  |  |  | <pre> +--ro direction? </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

```

|                                     te-label-direction
+--ro protection-type?               identityref
+--ro tunnel-termination-points
|   +--ro source?                    binary
|   +--ro destination?               binary
+--ro tunnels
|   +--ro sharing?                   boolean
|   +--ro tunnel* [tunnel-name]
|       +--ro tunnel-name            string
|       +--ro sharing?               boolean
+--ro path-constraints
|   +--ro te-bandwidth
|       +--ro (technology)?
|           +--:(generic)
|               +--ro generic?       te-bandwidth
+--ro link-protection?               identityref
+--ro setup-priority?                uint8
+--ro hold-priority?                 uint8
+--ro signaling-type?                identityref
+--ro path-metric-bounds
|   +--ro path-metric-bound* [metric-type]
|       +--ro metric-type            identityref
|       +--ro upper-bound?           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 disjointness?                  te-path-disjointness

```

```

+--ro optimizations
  +--ro (algorithm)?
    +--:(metric) {path-optimization-metric}?
      +--ro optimization-metric* [metric-type]
        +--ro metric-type
          | identityref
        +--ro weight?
          | uint8
        +--ro explicit-route-exclude-objects
          +--ro route-object-exclude-object*
            [index]
          +--ro index
            | uint32
          +--ro (type)?
            +--:(numbered-node-hop)
              +--ro numbered-node-hop
                +--ro node-id      te-node-id
                +--ro hop-type?   te-hop-type
            +--:(numbered-link-hop)
              +--ro numbered-link-hop
                +--ro link-tp-id   te-tp-id
                +--ro hop-type?
                | te-hop-type
                +--ro direction?
                  te-link-direction
            +--:(unnumbered-link-hop)
              +--ro unnumbered-link-hop
                +--ro link-tp-id   te-tp-id
                +--ro node-id
                | te-node-id
                +--ro hop-type?
                | te-hop-type
                +--ro direction?
                  te-link-direction
            +--:(as-number)
              +--ro as-number-hop
                +--ro as-number
                | inet:as-number
                +--ro hop-type?
                  te-hop-type
            +--:(label)

```

|                         |  |  |  |                                      |                     |
|-------------------------|--|--|--|--------------------------------------|---------------------|
|                         |  |  |  |                                      | +--ro label-hop     |
|                         |  |  |  |                                      | +--ro te-label      |
|                         |  |  |  |                                      | +--ro (technology)? |
|                         |  |  |  |                                      | +--:(generic)       |
|                         |  |  |  |                                      | +--ro generic?      |
|                         |  |  |  |                                      | rt-                 |
| types:generalized-label |  |  |  |                                      |                     |
|                         |  |  |  |                                      | +--ro direction?    |
|                         |  |  |  |                                      | te-label-direction  |
|                         |  |  |  | +--:(srlg)                           |                     |
|                         |  |  |  | +--ro srlg                           |                     |
|                         |  |  |  | +--ro srlg?                          | uint32              |
|                         |  |  |  | +--ro explicit-route-include-objects |                     |
|                         |  |  |  | +--ro route-object-include-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                      |                     |

```

types:generalized-label
    inet:as-number
    +---ro hop-type?
        te-hop-type
    +---:(label)
        +---ro label-hop
            +---ro te-label
                +---ro (technology)?
                    +---:(generic)
                        +---ro generic?
                            rt-
                                +---ro direction?
                                    te-label-direction
                                +---ro tiebreakers
                                    +---ro tiebreaker* [tiebreaker-type]
  +---ro tiebreaker-type      identityref
    +---:(objective-function)
        {path-optimization-objective-function}?
        +---ro objective-function
            +---ro objective-function-type?      identityref
+---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

```

|       |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------|--|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  | <pre> +--ro path-route-objects   +--ro path-route-object* [index]     +--ro index                               uint32     +--ro (type)?       +--:(numbered-node-hop)         +--ro numbered-node-hop           +--ro node-id       te-node-id           +--ro hop-type?     te-hop-type       +--:(numbered-link-hop)         +--ro numbered-link-hop           +--ro link-tp-id     te-tp-id           +--ro hop-type?     te-hop-type           +--ro direction?    te-link-direction       +--:(unnumbered-link-hop)         +--ro unnumbered-link-hop           +--ro link-tp-id     te-tp-id           +--ro node-id       te-node-id           +--ro hop-type?     te-hop-type           +--ro direction?    te-link-direction       +--:(as-number)         +--ro as-number-hop           +--ro as-number     inet:as-number           +--ro hop-type?     te-hop-type       +--:(label)         +--ro label-hop           +--ro te-label             +--ro (technology)?               +--:(generic)                 +--ro generic?                   rt-types:generalized- </pre> |
| label |  |  | <pre>           +--ro direction?             te-label-direction +--ro connectivity-matrix* [id]   +--ro id                               uint32   +--ro from     +--ro tp-ref?                       leafref     +--ro label-restrictions       +--ro label-restriction* [index]         +--ro restriction?             enumeration         +--ro index                     uint32 </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|       |  |  |                                                                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                |
|-------|--|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  |                                                                                                                                                                                                                                                                                                                                                                     | <pre> +--ro label-start     +--ro te-label         +--ro (technology)?             +--:(generic)                 +--ro generic?                     rt-types:generalized- </pre>                                               |
| label |  |  |                                                                                                                                                                                                                                                                                                                                                                     | <pre>     +--ro direction?         te-label-direction +--ro label-end     +--ro te-label         +--ro (technology)?             +--:(generic)                 +--ro generic?                     rt-types:generalized- </pre> |
| label |  |  |                                                                                                                                                                                                                                                                                                                                                                     | <pre>     +--ro direction?         te-label-direction +--ro label-step     +--ro (technology)?         +--:(generic)             +--ro generic?   int32 +--ro range-bitmap?   yang:hex-string </pre>                           |
|       |  |  | <pre> +--ro to +--ro tp-ref?           leafref +--ro label-restrictions     +--ro label-restriction* [index]         +--ro restriction?   enumeration         +--ro index         uint32 +--ro label-start     +--ro te-label         +--ro (technology)?             +--:(generic)                 +--ro generic?                     rt-types:generalized- </pre> |                                                                                                                                                                                                                                |
| label |  |  |                                                                                                                                                                                                                                                                                                                                                                     | <pre>     +--ro direction?         te-label-direction +--ro label-end     +--ro te-label         +--ro (technology)? </pre>                                                                                                    |



|       |  |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------|--|--|--|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  |  |  | <pre>       +---:(generic)         +---ro generic?           rt-types:generalized- </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| label |  |  |  |  | <pre>       +---ro direction?         te-label-direction       +---ro label-step       +---ro (technology)?         +---:(generic)           +---ro generic?    int32           +---ro range-bitmap? yang:hex-string       +---ro is-allowed?      boolean       +---ro underlay {te-topology-hierarchy}?         +---ro enabled?      boolean         +---ro primary-path           +---ro network-ref?             -&gt; /nw:networks/network/network-id           +---ro path-element* [path-element-id]             +---ro path-element-id      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 </pre> |

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| types:generalized-label | <pre> +--:(label)   +--ro label-hop     +--ro te-label       +--ro (technology)?         +--:(generic)           +--ro generic?             rt-            +--ro direction?             te-label-direction +--ro backup-path* [index]   +--ro index          uint32   +--ro network-ref?       -&gt; /nw:networks/network/network-id +--ro path-element* [path-element-id]   +--ro path-element-id          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 </pre> |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

```

types:generalized-label
    +--ro (technology)?
    +--:(generic)
    +--ro generic?
    rt-
    +--ro direction?
    te-label-direction
    +--ro protection-type?
    identityref
    +--ro tunnel-termination-points
    |   +--ro source?
    |   binary
    |   +--ro destination?
    |   binary
    +--ro tunnels
    |   +--ro sharing?
    |   boolean
    |   +--ro tunnel* [tunnel-name]
    |   |   +--ro tunnel-name
    |   |   string
    |   |   +--ro sharing?
    |   |   boolean
    +--ro path-constraints
    +--ro te-bandwidth
    |   +--ro (technology)?
    |   +--:(generic)
    |   |   +--ro generic?
    |   |   te-bandwidth
    +--ro link-protection?
    identityref
    +--ro setup-priority?
    uint8
    +--ro hold-priority?
    uint8
    +--ro signaling-type?
    identityref
    +--ro path-metric-bounds
    |   +--ro path-metric-bound* [metric-type]
    |   |   +--ro metric-type
    |   |   identityref
    |   |   +--ro upper-bound?
    |   |   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 disjointness?
|         te-path-disjointness
+---ro optimizations
+---ro (algorithm)?
|         +---:(metric) {path-optimization-metric}?
|         |         +---ro optimization-metric* [metric-type]
|         |         |         +---ro metric-type
|         |         |         |         identityref
|         |         |         +---ro weight?
|         |         |         |         uint8
|         |         |         +---ro explicit-route-exclude-objects
|         |         |         |         +---ro route-object-exclude-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

```

[illegible]



```

|   +--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:generalized-label
    +--ro direction?
      te-label-direction
    +--ro domain-id?
      uint32
    +--ro is-abstract?
      empty
    +--ro name?
      string
    +--ro signaling-address*
      inet:ip-address
    +--ro underlay-topology {te-topology-hierarchy}?
      +--ro network-ref?  -> /nw:networks/network/network-id
+--ro statistics
  +--ro discontinuity-time?
    yang:date-and-time
  +--ro node
    +--ro disables?
      yang:counter32
    +--ro enables?
      yang:counter32
    +--ro maintenance-sets?
      yang:counter32
    +--ro maintenance-clears?
      yang:counter32
    +--ro modifies?
      yang:counter32
  +--ro connectivity-matrix-entry
    +--ro creates?
      yang:counter32
    +--ro deletes?
      yang:counter32
    +--ro disables?
      yang:counter32
    +--ro enables?
      yang:counter32
    +--ro modifies?
      yang:counter32
+--rw tunnel-termination-point* [tunnel-tp-id]
  +--rw tunnel-tp-id
    binary
  +--rw admin-status?
    |
    te-types:te-admin-status
  +--rw name?
    string
  +--rw switching-capability?
    identityref
  +--rw encoding?
    identityref
  +--rw inter-layer-lock-id*
    uint32

```



```

+--rw protection-type?                                identityref
+--rw client-layer-adaptation
|   +--rw switching-capability*
|   |   [switching-capability encoding]
|   |   +--rw switching-capability    identityref
|   |   +--rw encoding                identityref
|   |   +--rw te-bandwidth
|   |   |   +--rw (technology)?
|   |   |   +--:(generic)
|   |   |   +--rw generic?    te-bandwidth
+--rw local-link-connectivities
|   +--rw number-of-entries?          uint16
|   +--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 is-allowed?                boolean
+--rw underlay {te-topology-hierarchy}?
|   +--rw enabled?                boolean
|   +--rw primary-path
|   |   +--rw network-ref?
|   |   |   -> /nw:networks/network/network-id

```

|       |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  | <pre> +--rw path-element* [path-element-id] +--rw path-element-id          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- </pre> |
| label |  |  | <pre> +--rw direction? +--rw te-label-direction +--rw backup-path* [index] +--rw index          uint32 +--rw network-ref?     -&gt; /nw:networks/network/network-id +--rw path-element* [path-element-id] +--rw path-element-id          uint32 +--rw (type)? +--: (numbered-node-hop)     +--rw numbered-node-hop </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

|       |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|--|--|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  |  | <pre>       +--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- </pre> |
| label |  |  |  | <pre>       +--rw direction?         te-label-direction +--rw protection-type?      identityref +--rw tunnel-termination-points   +--rw source?             binary   +--rw destination?        binary +--rw tunnels   +--rw sharing?            boolean   +--rw tunnel* [tunnel-name]     +--rw tunnel-name       string     +--rw sharing?          boolean +--rw path-constraints   +--rw te-bandwidth     +--rw (technology)?       +--:(generic)         +--rw generic?     te-bandwidth </pre>                                                                                                                                                                                                                                                  |

```

+--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 optimizations
|   +--rw (algorithm)?
|   |   +--:(metric) {path-optimization-metric}?
|   |   |   +--rw optimization-metric* [metric-type]
|   |   |   |   +--rw metric-type
|   |   |   |   |   identityref
|   |   |   |   +--rw weight?
|   |   |   |   |   uint8
|   |   |   +--rw explicit-route-exclude-objects
|   |   |   |   +--rw route-object-exclude-object*
|   |   |   |   |   [index]
|   |   |   |   |   +--rw index
|   |   |   |   |   |   uint32
|   |   |   +--rw (type)?
|   |   |   |   +--:(numbered-node-hop)

```

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

|  |  |  |  |                                     |
|--|--|--|--|-------------------------------------|
|  |  |  |  | +--rw index                         |
|  |  |  |  | uint32                              |
|  |  |  |  | +--rw (type)?                       |
|  |  |  |  | +--:(numbered-node-hop)             |
|  |  |  |  | +--rw numbered-node-hop             |
|  |  |  |  | +--rw node-id     te-node-id        |
|  |  |  |  | +--rw hop-type?   te-hop-type       |
|  |  |  |  | +--:(numbered-link-hop)             |
|  |  |  |  | +--rw numbered-link-hop             |
|  |  |  |  | +--rw link-tp-id   te-tp-id         |
|  |  |  |  | +--rw hop-type?                     |
|  |  |  |  | te-hop-type                         |
|  |  |  |  | +--rw direction?                    |
|  |  |  |  | te-link-direction                   |
|  |  |  |  | +--:(unnumbered-link-hop)           |
|  |  |  |  | +--rw unnumbered-link-hop           |
|  |  |  |  | +--rw link-tp-id   te-tp-id         |
|  |  |  |  | +--rw node-id                       |
|  |  |  |  | te-node-id                          |
|  |  |  |  | +--rw hop-type?                     |
|  |  |  |  | te-hop-type                         |
|  |  |  |  | +--rw direction?                    |
|  |  |  |  | te-link-direction                   |
|  |  |  |  | +--:(as-number)                     |
|  |  |  |  | +--rw as-number-hop                 |
|  |  |  |  | +--rw as-number                     |
|  |  |  |  | inet:as-number                      |
|  |  |  |  | +--rw hop-type?                     |
|  |  |  |  | te-hop-type                         |
|  |  |  |  | +--:(label)                         |
|  |  |  |  | +--rw label-hop                     |
|  |  |  |  | +--rw te-label                      |
|  |  |  |  | +--rw (technology)?                 |
|  |  |  |  | +--:(generic)                       |
|  |  |  |  | +--rw generic?                      |
|  |  |  |  | rt-                                 |
|  |  |  |  | +--rw direction?                    |
|  |  |  |  | te-label-direction                  |
|  |  |  |  | +--rw tiebreakers                   |
|  |  |  |  | +--rw tiebreaker* [tiebreaker-type] |

types:generalized-label

```

|         +--rw tiebreaker-type      identityref
+--:(objective-function)
|         {path-optimization-objective-function}?
|         +--rw objective-function
|         +--rw objective-function-type?  identityref
+--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:generalized-

label
+--ro direction?
te-label-direction
+--rw local-link-connectivity* [link-tp-ref]
+--rw link-tp-ref
|
-> ../../../../nt:termination-point/tp-id
+--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 is-allowed?      boolean
+---rw underlay {te-topology-hierarchy}?
|   +---rw enabled?          boolean
|   +---rw primary-path
|   |   +---rw network-ref?
|   |   |   -> /nw:networks/network/network-id
|   +---rw path-element* [path-element-id]
|   |   +---rw path-element-id          uint32
|   +---rw (type)?
|   |   +---:(numbered-node-hop)
|   |   |   +---rw numbered-node-hop
|   |   |   |   +---rw node-id      te-node-id
|   |   |   |   +---rw hop-type?    te-hop-type
|   |   +---:(numbered-link-hop)
|   |   |   +---rw numbered-link-hop
|   |   |   |   +---rw link-tp-id    te-tp-id
|   |   |   |   +---rw hop-type?    te-hop-type
|   |   |   |   +---rw direction?
|   |   |   |   |   te-link-direction
|   |   +---:(unnumbered-link-hop)
|   |   |   +---rw unnumbered-link-hop
|   |   |   |   +---rw link-tp-id    te-tp-id
|   |   |   |   +---rw node-id      te-node-id
|   |   |   |   +---rw hop-type?    te-hop-type
|   |   |   |   +---rw direction?
|   |   |   |   |   te-link-direction
|   +---:(as-number)
|   |   +---rw as-number-hop
|   |   |   +---rw as-number      inet:as-number
|   |   |   +---rw hop-type?      te-hop-type
|   +---:(label)
|   |   +---rw label-hop
|   |   |   +---rw te-label
|   |   |   |   +---rw (technology)?
|   |   |   |   |   +---:(generic)
|   |   |   |   |   |   +---rw generic?
|   |   |   |   |   |   |   rt-
types:generalized-label

```

```

|                                     +--rw direction?
|                                     te-label-direction
+--rw backup-path* [index]
|   +--rw index                      uint32
|   +--rw network-ref?
|   |   -> /nw:networks/network/network-id
+--rw path-element* [path-element-id]
|   +--rw path-element-id            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-
|   |   |   |   |   |   |   +--rw direction?
|   |   |   |   |   |   |   te-label-direction
+--rw protection-type?              identityref

```

types:generalized-label

```

+--rw tunnel-termination-points
|   +--rw source?      binary
|   +--rw destination? binary
+--rw tunnels
|   +--rw sharing?     boolean
|   +--rw tunnel* [tunnel-name]
|       +--rw tunnel-name    string
|       +--rw sharing?      boolean
+--rw path-constraints
+--rw te-bandwidth
|   +--rw (technology)?
|       +--:(generic)
|           +--rw generic?    te-bandwidth
+--rw link-protection?      identityref
+--rw setup-priority?       uint8
+--rw hold-priority?        uint8
+--rw signaling-type?       identityref
+--rw path-metric-bounds
|   +--rw path-metric-bound* [metric-type]
|   +--rw metric-type        identityref
|   +--rw upper-bound?      uint64
+--rw path-affinities-values
|   +--rw path-affinities-value* [usage]
|   +--rw usage              identityref
|   +--rw value?            admin-groups
+--rw path-affinity-names
|   +--rw path-affinity-name* [usage]
|   +--rw usage              identityref
|   +--rw affinity-name* [name]
|   +--rw name               string
+--rw path-srlgs-lists
|   +--rw path-srlgs-list* [usage]
|   +--rw usage              identityref
|   +--rw values*           srlg
+--rw path-srlgs-names
|   +--rw path-srlgs-name* [usage]
|   +--rw usage              identityref
|   +--rw names*            string
+--rw disjointness?
|   te-path-disjointness
+--rw optimizations

```

```

+--rw (algorithm)?
  +--:(metric) {path-optimization-metric}?
    +--rw optimization-metric* [metric-type]
      +--rw metric-type
        | identityref
      +--rw weight?
        | uint8
      +--rw explicit-route-exclude-objects
        +--rw route-object-exclude-object*
          [index]
        +--rw index
          | uint32
        +--rw (type)?
          +--:(numbered-node-hop)
            +--rw numbered-node-hop
              +--rw node-id
                | te-node-id
              +--rw hop-type?
                | te-hop-type
          +--:(numbered-link-hop)
            +--rw numbered-link-hop
              +--rw link-tp-id
                | te-tp-id
              +--rw hop-type?
                | te-hop-type
              +--rw direction?
                | te-link-direction
          +--:(unnumbered-link-hop)
            +--rw unnumbered-link-hop
              +--rw link-tp-id
                | te-tp-id
              +--rw node-id
                | te-node-id
              +--rw hop-type?
                | te-hop-type
              +--rw direction?
                | te-link-direction
          +--:(as-number)
            +--rw as-number-hop
              +--rw as-number
                | inet:as-number

```

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

|                         |  |  |  |  |  |                                      |
|-------------------------|--|--|--|--|--|--------------------------------------|
|                         |  |  |  |  |  | te-node-id                           |
|                         |  |  |  |  |  | +--rw hop-type?                      |
|                         |  |  |  |  |  | te-hop-type                          |
|                         |  |  |  |  |  | +--rw direction?                     |
|                         |  |  |  |  |  | te-link-direction                    |
|                         |  |  |  |  |  | +---:(as-number)                     |
|                         |  |  |  |  |  | +--rw as-number-hop                  |
|                         |  |  |  |  |  | +--rw as-number                      |
|                         |  |  |  |  |  | inet:as-number                       |
|                         |  |  |  |  |  | +--rw hop-type?                      |
|                         |  |  |  |  |  | te-hop-type                          |
|                         |  |  |  |  |  | +---:(label)                         |
|                         |  |  |  |  |  | +--rw label-hop                      |
|                         |  |  |  |  |  | +--rw te-label                       |
|                         |  |  |  |  |  | +--rw (technology)?                  |
|                         |  |  |  |  |  | +---:(generic)                       |
|                         |  |  |  |  |  | +--rw generic?                       |
|                         |  |  |  |  |  | rt-                                  |
| types:generalized-label |  |  |  |  |  |                                      |
|                         |  |  |  |  |  | +--rw direction?                     |
|                         |  |  |  |  |  | te-label-                            |
| direction               |  |  |  |  |  |                                      |
|                         |  |  |  |  |  | +--rw tiebreakers                    |
|                         |  |  |  |  |  | +--rw tiebreaker* [tiebreaker-type]  |
|                         |  |  |  |  |  | +--rw tiebreaker-type identityref    |
|                         |  |  |  |  |  | +---:(objective-function)            |
|                         |  |  |  |  |  | {path-optimization-objective-        |
| function}?              |  |  |  |  |  |                                      |
|                         |  |  |  |  |  | +--rw objective-function             |
|                         |  |  |  |  |  | +--rw objective-function-type?       |
|                         |  |  |  |  |  | identityref                          |
|                         |  |  |  |  |  | +--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:generalized-label
|                                     | +--ro direction?
|                                     | te-label-direction
| +--ro oper-status?
| | te-types:te-oper-status
| +--ro geolocation
| | +--ro altitude?      int64
| | +--ro latitude?     geographic-coordinate-degree
| | +--ro longitude?    geographic-coordinate-degree
| +--ro statistics
| | +--ro discontinuity-time?      yang:date-and-time
| | +--ro tunnel-termination-point
| | | +--ro disables?      yang:counter32
| | | +--ro enables?      yang:counter32
| | | +--ro maintenance-clears? yang:counter32
| | | +--ro maintenance-sets? yang:counter32
| | | +--ro modifies?      yang:counter32
| | | +--ro downs?        yang:counter32
| | | +--ro ups?          yang:counter32
| | | +--ro in-service-clears? yang:counter32
| | | +--ro in-service-sets? yang:counter32
| | +--ro local-link-connectivity
| | | +--ro creates?      yang:counter32
| | | +--ro deletes?     yang:counter32
| | | +--ro disables?    yang:counter32
| | | +--ro enables?     yang:counter32
| | | +--ro modifies?    yang:counter32
| +--rw supporting-tunnel-termination-point*
| | [node-ref tunnel-tp-ref]
| | +--rw node-ref      inet:uri
| | +--rw tunnel-tp-ref binary
augment /nw:networks/nw:network/nt:link:
+--rw te!
| +--rw (bundle-stack-level)?
| | +--:(bundle)
| | | +--rw bundled-links
| | | | +--rw bundled-link* [sequence]
| | | | | +--rw sequence      uint32
| | | | | +--rw src-tp-ref?   leafref

```



```

|         +---rw des-tp-ref?   leafref
+---:(component)
|         +---rw component-links
|         |         +---rw component-link* [sequence]
|         |         |         +---rw sequence           uint32
|         |         |         +---rw src-interface-ref?  string
|         |         |         +---rw des-interface-ref?  string
+---rw te-link-template*
|         -> ../../../../te/templates/link-template/name
|         {template}?
+---rw te-link-attributes
|         +---rw access-type?
|         |         te-types:te-link-access-type
+---rw external-domain
|         +---rw network-ref?
|         |         -> /nw:networks/network/network-id
+---rw remote-te-node-id?      te-types:te-node-id
|         +---rw remote-te-link-tp-id?  te-types:te-tp-id
+---rw is-abstract?           empty
+---rw name?                  string
+---rw underlay {te-topology-hierarchy}?
|         +---rw enabled?          boolean
+---rw primary-path
|         +---rw network-ref?
|         |         -> /nw:networks/network/network-id
+---rw path-element* [path-element-id]
|         +---rw path-element-id      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

```

|       |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------|--|--|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |  |  |  | <pre>       +--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 +--rw backup-path* [index] +--rw index          uint32 +--rw network-ref?     -&gt; /nw:networks/network/network-id +--rw path-element* [path-element-id] +--rw path-element-id          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 </pre> |

```

|         +--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 protection-type?      identityref
+---rw tunnel-termination-points
|   +--rw source?            binary
|   +--rw destination?      binary
+---rw tunnels
|   +--rw sharing?          boolean
|   +--rw tunnel* [tunnel-name]
|   |   +--rw tunnel-name    string
|   |   +--rw sharing?      boolean
+---rw admin-status?
|   te-types:te-admin-status
+---rw link-index?            uint64
+---rw administrative-group?
|   te-types:admin-groups
+---rw interface-switching-capability*
|   [switching-capability encoding]
|   +--rw switching-capability  identityref
|   +--rw encoding              identityref
|   +--rw max-lsp-bandwidth* [priority]
|   |   +--rw priority          uint8
|   |   +--rw te-bandwidth
|   |   |   +--rw (technology)?
|   |   |   +---:(generic)
|   |   |   +--rw generic?    te-bandwidth
+---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 link-protection-type?    identityref
+---rw max-link-bandwidth
|   +---rw te-bandwidth
|   +---rw (technology)?
|   |   +---:(generic)
|   |       +---rw generic?    te-bandwidth
+---rw max-resv-link-bandwidth
|   +---rw te-bandwidth
|   +---rw (technology)?
|   |   +---:(generic)
|   |       +---rw generic?    te-bandwidth
+---rw unreserved-bandwidth* [priority]
|   +---rw priority          uint8
|   +---rw te-bandwidth
|   +---rw (technology)?
|   |   +---:(generic)
|   |       +---rw generic?    te-bandwidth
+---rw te-default-metric?          uint32
+---rw te-delay-metric?            uint32
+---rw te-igp-metric?             uint32
+---rw te-srlgs
|   +---rw value*    te-types:srlg
+---rw te-nsrlgs {nsrlg}?
|   +---rw id*    uint32

```

```

+--ro oper-status?                te-types:te-oper-status
+--ro is-transitional?            empty
+--ro information-source?         te-info-source
+--ro information-source-instance? string
+--ro information-source-state
|   +--ro credibility-preference? uint16
|   +--ro logical-network-element? string
|   +--ro network-instance?       string
|   +--ro topology
|       +--ro link-ref?           leafref
|       +--ro network-ref?       -> /nw:networks/network/network-id
+--ro information-source-entry*
|   [information-source information-source-instance]
|   +--ro information-source         te-info-source
|   +--ro information-source-instance string
|   +--ro information-source-state
|       +--ro credibility-preference? uint16
|       +--ro logical-network-element? string
|       +--ro network-instance?       string
|       +--ro topology
|           +--ro link-ref?           leafref
|           +--ro network-ref?       -> /nw:networks/network/network-id
+--ro link-index?                 uint64
+--ro administrative-group?
|   te-types:admin-groups
+--ro interface-switching-capability*
|   [switching-capability encoding]
|   +--ro switching-capability      identityref
|   +--ro encoding                  identityref
|   +--ro max-lsp-bandwidth* [priority]
|       +--ro priority              uint8
|       +--ro te-bandwidth
|           +--ro (technology)?
|               +--:(generic)
|                   +--ro generic?   te-bandwidth
+--ro label-restrictions
|   +--ro label-restriction* [index]
|       +--ro restriction?      enumeration
|       +--ro index              uint32
|       +--ro label-start

```

```

    +--ro te-label
      +--ro (technology)?
        +--:(generic)
          +--ro generic?
            rt-types:generalized-label
      +--ro direction?          te-label-direction
    +--ro label-end
      +--ro te-label
        +--ro (technology)?
          +--:(generic)
            +--ro generic?
              rt-types:generalized-label
        +--ro direction?          te-label-direction
    +--ro label-step
      +--ro (technology)?
        +--:(generic)
          +--ro generic?    int32
    +--ro range-bitmap?    yang:hex-string
+--ro link-protection-type?    identityref
+--ro max-link-bandwidth
  +--ro te-bandwidth
    +--ro (technology)?
      +--:(generic)
        +--ro generic?    te-bandwidth
+--ro max-resv-link-bandwidth
  +--ro te-bandwidth
    +--ro (technology)?
      +--:(generic)
        +--ro generic?    te-bandwidth
+--ro unreserved-bandwidth* [priority]
  +--ro priority          uint8
  +--ro te-bandwidth
    +--ro (technology)?
      +--:(generic)
        +--ro generic?    te-bandwidth
+--ro te-default-metric?          uint32
+--ro te-delay-metric?            uint32
+--ro te-igp-metric?             uint32
+--ro te-srlgs
  +--ro value*    te-types:srlg
+--ro te-nsrlgs {nsrlg}?

```

```

    |         +---ro id*      uint32
+---ro recovery
    |         +---ro restoration-status?   te-types:te-recovery-status
    |         +---ro protection-status?    te-types:te-recovery-status
+---ro underlay {te-topology-hierarchy}?
    |         +---ro dynamic?      boolean
    |         +---ro committed?    boolean
+---ro statistics
    |         +---ro discontinuity-time?    yang:date-and-time
    |         +---ro disables?             yang:counter32
    |         +---ro enables?             yang:counter32
    |         +---ro maintenance-clears?   yang:counter32
    |         +---ro maintenance-sets?    yang:counter32
    |         +---ro modifies?            yang:counter32
    |         +---ro downs?               yang:counter32
    |         +---ro ups?                 yang:counter32
    |         +---ro fault-clears?        yang:counter32
    |         +---ro fault-detects?       yang:counter32
    |         +---ro protection-switches?  yang:counter32
    |         +---ro protection-reverts?   yang:counter32
    |         +---ro restoration-failures? yang:counter32
    |         +---ro restoration-starts?   yang:counter32
    |         +---ro restoration-successes? yang:counter32
    |         +---ro restoration-reversion-failures? yang:counter32
    |         +---ro restoration-reversion-starts? yang:counter32
    |         +---ro restoration-reversion-successes? yang:counter32
augment /nw:networks/nw:network/nw:node/nt:termination-point:
+---rw te-tp-id?   te-types:te-tp-id
+---rw te!
    +---rw admin-status?
    |         te-types:te-admin-status
    +---rw name?                                string
    +---rw interface-switching-capability*
    |         [switching-capability encoding]
    |         +---rw switching-capability      identityref
    |         +---rw encoding                  identityref
    |         +---rw max-lsp-bandwidth* [priority]
    |         |         +---rw priority          uint8
    |         |         +---rw te-bandwidth
    |         |         |         +---rw (technology)?
    |         |         |         |         +---:(generic)

```

```
|          +--rw generic?    te-bandwidth
+--rw inter-domain-plug-id?          binary
+--rw inter-layer-lock-id*           uint32
+--ro oper-status?
|   te-types:te-oper-status
+--ro geolocation
    +--ro altitude?    int64
    +--ro latitude?    geographic-coordinate-degree
    +--ro longitude?   geographic-coordinate-degree
```



## Appendix B. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module `ietf-te-topology` defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module `ietf-te-topology-state` is defined as a state model, which mirrors the module `ietf-te-topology` defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion module, `ietf-te-topology-state`, is redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the module `ietf-te-topology-state` mirrors that of the module `ietf-te-topology`. The YANG tree of the module `ietf-te-topology-state` is not depicted separately.

## B.1. TE Topology State YANG Module

This module references [RFC6001], [RFC8345], and [I-D.ietf-teas-yang-te-types].

```
<CODE BEGINS> file "ietf-te-topology-state@2019-02-07.yang"
module ietf-te-topology-state {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-state";

  prefix "tet-s";

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

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

  import ietf-network-state {
```

```
    prefix "nw-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
}

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

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

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

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

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

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

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

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

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

description
    "TE topology state model.

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

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

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

```
revision "2019-02-07" {
  description "Initial revision";
  reference "RFC XXXX: YANG Data Model for TE Topologies";
  // RFC Ed.: replace XXXX with actual RFC number and remove
  // this note
}

/*
 * Groupings
 */
grouping te-node-connectivity-matrix-attributes {
  description
    "Termination point references of a connectivity matrix entry.";
  container from {
    description
      "Reference to source link termination point.";
    leaf tp-ref {
      type leafref {
        path "../..//..//..//..//nt-s:termination-point/nt-s:tp-id";
      }
      description
        "Relative reference to a termination point.";
    }
    uses te-types:label-set-info;
  }
  container to {
    description
      "Reference to destination link termination point.";
    leaf tp-ref {
      type leafref {
        path "../..//..//..//..//nt-s:termination-point/nt-s:tp-id";
      }
    }
  }
}
```

```
    }
    description
      "Relative reference to a termination point.";
  }
  uses te-types:label-set-info;
}
uses tet:connectivity-matrix-entry-path-attributes;
} // te-node-connectivity-matrix-attributes

grouping te-node-tunnel-termination-point-llc-list {
  description
    "Local link connectivity list of a tunnel termination
     point on a TE node.";
  list local-link-connectivity {
    key "link-tp-ref";
    description
      "The termination capabilities between
       tunnel-termination-point and link termination-point.
       The capability information can be used to compute
       the tunnel path.
       The Interface Adjustment Capability Descriptors (IACD)
       (defined in RFC 6001) on each link-tp can be derived from
       this local-link-connectivity list.";
    reference
      "RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
       for Multi-Layer and Multi-Region Networks (MLN/MRN).";

    leaf link-tp-ref {
      type leafref {
        path "../.../.../nt-s:termination-point/nt-s:tp-id";
      }
      description
        "Link termination point.";
    }
    uses te-types:label-set-info;
    uses tet:connectivity-matrix-entry-path-attributes;
  } // local-link-connectivity
} // te-node-tunnel-termination-point-config

/*
 * Data nodes
```

```
*/
augment "/nw-s:networks/nw-s:network/nw-s:network-types" {
  description
    "Introduce new network type for TE topology.";
  container te-topology {
    presence "Indicates TE topology.";
    description
      "Its presence identifies the TE topology type.";
  }
}

augment "/nw-s:networks" {
  description
    "Augmentation parameters for TE topologies.";
  uses tet:te-topologies-augment;
}

augment "/nw-s:networks/nw-s:network" {
  when "nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Configuration parameters for TE topology.";
  uses tet:te-topology-augment;
}

augment "/nw-s:networks/nw-s:network/nw-s:node" {
  when "../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Configuration parameters for TE at node level.";
  leaf te-node-id {
    type te-types:te-node-id;
    description
      "The identifier of a node in the TE topology.
      A node is specific to a topology to which it belongs.";
  }
}
```

```
    }
    container te {
      must "../te-node-id" {
        description
          "te-node-id is mandatory.";
      }
      must "count(..nw-s:supporting-node)<=1" {
        description
          "For a node in a TE topology, there cannot be more
           than 1 supporting node. If multiple nodes are abstracted,
           the underlay-topology is used.";
      }
      presence "TE support.";
      description
        "Indicates TE support.";
      uses tet:te-node-augment;
    } // te
  }

  augment "/nw-s:networks/nw-s:network/nt-s:link" {
    when "../nw-s:network-types/tet-s:te-topology" {
      description
        "Augmentation parameters apply only for networks with
         TE topology type.";
    }
    description
      "Configuration parameters for TE at link level.";
    container te {
      must "count(..nt-s:supporting-link)<=1" {
        description
          "For a link in a TE topology, there cannot be more
           than 1 supporting link. If one or more link paths are
           abstracted, the underlay is used.";
      }
      presence "TE support.";
      description
        "Indicates TE support.";
      uses tet:te-link-augment;
    } // te
  }
}
```

```
augment "/nw-s:networks/nw-s:network/nw-s:node/"
  + "nt-s:termination-point" {
    when "../..nw-s:network-types/tet-s:te-topology" {
      description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
      "Configuration parameters for TE at termination point level.";
    uses tet:te-termination-point-augment;
  }

augment
  "/nw-s:networks/nw-s:network/nt-s:link/te/bundle-stack-level/"
  + "bundle/bundled-links/bundled-link" {
    when "../..nw-s:network-types/tet-s:te-topology" {
      description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
      "Augment TE link bundled link.";
    leaf src-tp-ref {
      type leafref {
        path "../..nw-s:node[nw-s:node-id = "
          + "current()../..nt-s:source/"
          + "nt-s:source-node]/"
          + "nt-s:termination-point/nt-s:tp-id";
        require-instance true;
      }
      description
        "Reference to another TE termination point on the
        same source node.";
    }
    leaf des-tp-ref {
      type leafref {
        path "../..nw-s:node[nw-s:node-id = "
          + "current()../..nt-s:destination/"
          + "nt-s:dest-node]/"
          + "nt-s:termination-point/nt-s:tp-id";
        require-instance true;
      }
    }
  }
```

```
    }
    description
      "Reference to another TE termination point on the
       same destination node.";
  }
}

augment
  "/nw-s:networks/nw-s:network/nw-s:node/te/"
+ "information-source-entry/connectivity-matrices/"
+ "connectivity-matrix" {
  when "../..../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
       TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
  "/nw-s:networks/nw-s:network/nw-s:node/te/te-node-attributes/"
+ "connectivity-matrices/connectivity-matrix" {
  when "../..../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
       TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
  "/nw-s:networks/nw-s:network/nw-s:node/te/"
+ "tunnel-termination-point/local-link-connectivities" {
  when "../..../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
       TE topology type.";
```



```
    }  
    description  
        "Augment TE node tunnel termination point LLCs  
        (Local Link Connectivities).";  
    uses te-node-tunnel-termination-point-llc-list;  
    }  
}  
<CODE ENDS>
```

## Appendix C. Example: YANG Model for Technology Specific Augmentations

This section provides an example YANG module to define a technology specific TE topology model for the example-topology described in Section 6.

```
module example-topology {
  yang-version 1.1;

  namespace "http://example.com/example-topology";
  prefix "ex-topo";

  import ietf-network {
    prefix "nw";
  }

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

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

  organization
    "Example Organization";
  contact
    "Editor: Example Author";

  description
    "This module defines a topology data model for the example
    technology.";

  revision 2018-06-15 {
    description
      "Initial revision.";
    reference
      "Example reference.";
  }

  /*
   * Data nodes
```

```
*/
augment "/nw:networks/nw:network/nw:network-types/"
+ "tet:te-topology" {
  description
    "Augment network types to define example topology type.";
  container example-topology {
    presence
      "Introduce new network type for example topology.";
    description
      "Its presence identifies the example topology type.";
  }
}

augment "/nw:networks/nw:network/tet:te" {
  when "../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  description "Augment network topology.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-1 {
      type uint8;
      description "Attribute 1 for example technology.";
    }
  }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes" {
  when "../..nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  description "Augment node attributes.";
  container attributes {
    description "Attributes for example technology.";
  }
}
```

```
    leaf attribute-2 {
      type uint8;
      description "Attribute 2 for example technology.";
    }
  }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices" {
  when "../..../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  description "Augment node connectivity matrices.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-3 {
      type uint8;
      description "Attribute 3 for example technology.";
    }
  }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix" {
  when "../..../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  description "Augment node connectivity matrix.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-3 {
      type uint8;
      description "Attribute 3 for example technology.";
    }
  }
}
```

```
    }
  }

  augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:tunnel-termination-point" {
    when "../.../nw:network-types/tet:te-topology/"
      + "ex-topo:example-topology" {
      description
        "Augmentation parameters apply only for networks with
        example topology type.";
    }
    description "Augment tunnel termination point.";
    container attributes {
      description "Attributes for example technology.";
      leaf attribute-4 {
        type uint8;
        description "Attribute 4 for example technology.";
      }
    }
  }
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/"
  + "tet:te" {
  when "../.../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  description "Augment link termination point.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-5 {
      type uint8;
      description "Attribute 5 for example technology.";
    }
  }
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:te-link-attributes" {
```

```
when "../../../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
description "Augment link attributes.";
container attributes {
  description "Attributes for example technology.";
  leaf attribute-6 {
    type uint8;
    description "Attribute 6 for example technology.";
  }
}
}

/*
 * Augment TE bandwidth.
 */

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:interface-switching-capability/tet:max-lsp-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:max-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  case "example" {
```

```
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:max-resv-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:unreserved-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
```

```

}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
when "../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf bandwidth-1 {
            type uint32;
            description "Bandwidth 1 for example technology.";
        }
    }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
when "../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf bandwidth-1 {
            type uint32;
            description "Bandwidth 1 for example technology.";
        }
    }
}
}

```



```

    }
    description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
when "../.../../.../../.../../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf bandwidth-1 {
type uint32;
description "Bandwidth 1 for example technology.";
}
}
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
when "../.../../.../../.../../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf bandwidth-1 {
type uint32;
description "Bandwidth 1 for example technology.";

```

```

    }
  }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:client-layer-adaptation/"
+ "tet:switching-capability/tet:te-bandwidth/tet:technology" {
  when "../../../../../../../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
  when "../../../../../../../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;

```

```
        description "Bandwidth 1 for example technology.";
    }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" {
when "../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf bandwidth-1 {
            type uint32;
            description "Bandwidth 1 for example technology.";
        }
    }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:interface-switching-capability/tet:max-lsp-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
when "../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example{
```

```
        description "Attributes for example technology.";
        leaf bandwidth-1 {
            type uint32;
            description "Bandwidth 1 for example technology.";
        }
    }
    description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:max-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
    when "../..../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf bandwidth-1 {
                type uint32;
                description "Bandwidth 1 for example technology.";
            }
        }
    }
    description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:max-resv-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
    when "../..../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
}
```

```
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf bandwidth-1 {
          type uint32;
          description "Bandwidth 1 for example technology.";
        }
      }
    }
    description "Augment TE bandwidth.";
  }

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:interface-switching-capability/tet:max-lsp-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  when "../..../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:max-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  when "../..../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
```

```
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:max-resv-link-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
  when "../..../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:unreserved-bandwidth/"
```

```
+ "tet:te-bandwidth/tet:technology" {
when "../..../..../..../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
}
description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/"
+ "tet:te/"
+ "tet:interface-switching-capability/tet:max-lsp-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
when "../..../..../..../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
}
description "Augment TE bandwidth.";
}
```

```
/*
 * Augment TE label.
 */

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  case "example" {
```



```

        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
}
description "Augment TE label.";
}

/* Under te-node-attributes/connectivity-matrices */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
    when "../..../nw:network-types/tet:te-topology/"
        + "ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {

```

```

        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../..../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
}

```

```

    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
    description "Augment TE label.";
  }

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../nw:network-types/"

```

```
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
         example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

/* Under te-node-attributes/.../connectivity-matrix */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:from/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
         example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}
```

```

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:from/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
  "Augmentation parameters apply only for networks with
  example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
  "Augmentation parameters apply only for networks with
  example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}

```

```

    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {

```

```

        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when "../..../nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when "../..../nw:network-types/"

```

```
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

/* Under information-source-entry/connectivity-matrices */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  when "../../../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
}
description "Augment TE label.";
}
```



```
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../.../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
"Augmentation parameters apply only for networks with
example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../.../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
"Augmentation parameters apply only for networks with
example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
}
}
```

```

    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../..../..../..../..../..../..../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../..../..../..../..../..../..../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;

```

```

        description "Label 1 for example technology.";
    }
}
description "Augment TE label.";
}

/* Under information-source-entry/.../connectivity-matrix */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:from/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:from/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with

```

```
        example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
    when "../..//../..//../..//../..//../..//nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
```

```

+ "tet:te-label/tet:technology" {
when "../..../..../..../..../..../..../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..../..../..../..../..../..../..../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

```

```
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
    }
  }
}
```

```
        description "Label 1 for example technology.";
    }
}
description "Augment TE label.";
}

/* Under tunnel-termination-point/local-link-connectivities */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../../../../../../../../../../../nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
}
```

```

case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description

```



```

        "Augmentation parameters apply only for networks with
        example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when "../..../nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
    description "Augment TE label.";
}

/* Under tunnel-termination-point/.../local-link-connectivity */

augment "/nw:networks/nw:network/nw:node/tet:te/"

```

```
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
}
description "Augment TE label.";}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
when "../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
container example {
description "Attributes for example technology.";
leaf label-1 {
type uint32;
description "Label 1 for example technology.";
}
}
```

```

    }
    description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
    }
}
}

```

```

        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:path-properties/tet:path-route-objects/"
+ "tet:path-route-object/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../..//../..//../..//../..//../..//../..//nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

/* Under te-link-attributes */

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
when "../..//../..//../..//../..//nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {

```

```
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
  when "../..//../..//../..//../nw:network-types/"
  + "tet:te-topology/ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
```

```
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
when "../../../../../../../../../../../../../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}
```

```
/* Under te-link information-source-entry */

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  when "../..//../..//../..//../nw:network-types/"
  + "tet:te-topology/ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:information-source-entry/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
  when "../..//../..//../..//../nw:network-types/"
  + "tet:te-topology/ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
}
```

```
    }  
  }  
  description "Augment TE label.";  
}  
}
```

#### Contributors

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

Dieter Beller  
Nokia  
Email: Dieter.Beller@nokia.com

Carlo Perocchio  
Ericsson  
Email: carlo.perocchio@ericsson.com

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

#### Authors' Addresses

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

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

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

Tarek Saad  
Juniper Networks  
Email: tsaad@juniper.net

Himanshu Shah  
Ciena  
Email: hshah@ciena.com



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



MPLS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 16 June 2022

K. Kompella  
R. Balaji  
Juniper Networks  
R. Thomas  
Cohesity  
13 December 2021

Label Distribution Using ARP  
draft-kompella-mpls-larp-11

Abstract

This document describes extensions to the Address Resolution Protocol to distribute MPLS labels for IPv4 and IPv6 host addresses. Distribution of labels via ARP enables simple plug-and-play operation of MPLS, which is key to deploying MPLS in data centers and enterprises.

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 16 June 2022.

Copyright Notice

Copyright (c) 2021 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 . . . . .              | 2  |
| 1.1. Requirements Language . . . . .   | 3  |
| 1.2. Approach . . . . .                | 3  |
| 2. Overview of Ethernet ARP . . . . .  | 3  |
| 3. L-ARP Protocol Operation . . . . .  | 4  |
| 3.1. Setup . . . . .                   | 5  |
| 3.2. Egress Operation . . . . .        | 5  |
| 3.3. Ingress Operation . . . . .       | 6  |
| 3.4. Data Plane . . . . .              | 6  |
| 4. Attributes . . . . .                | 7  |
| 4.1. Secondary Attributes . . . . .    | 7  |
| 5. L-ARP Message Format . . . . .      | 8  |
| 5.1. Hardware Address Format . . . . . | 9  |
| 5.2. CT TLV . . . . .                  | 9  |
| 6. Security Considerations . . . . .   | 10 |
| 7. IANA Considerations . . . . .       | 10 |
| 8. Acknowledgments . . . . .           | 10 |
| 9. References . . . . .                | 10 |
| 9.1. Normative References . . . . .    | 10 |
| 9.2. Informative References . . . . .  | 11 |
| Authors' Addresses . . . . .           | 12 |

## 1. Introduction

This document describes extensions to the Address Resolution Protocol (ARP) [RFC0826] to advertise label bindings for IP host addresses. While there are well-established protocols, such as LDP [RFC5036], RSVP [RFC3209], BGP [RFC3107] and SPRING-MPLS [RFC8660], that provide robust mechanisms for label distribution, these protocols tend to be relatively complex, and often require detailed configuration for proper operation. There are situations where a simpler protocol may be more suitable from an operational standpoint. An example is the case where an MPLS Fabric is the underlay technology in a Data Center; here, MPLS tunnels originate from host machines. The host thus needs a mechanism to acquire label bindings to participate in the MPLS Fabric, but in a simple, plug-and-play manner. Existing signaling/routing protocols do not always meet this need. Labeled

ARP (L-ARP) is a proposal to fill that gap.

### 1.1. 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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The term "server" will be used in this document to refer to an ARP/L-ARP server; the term "host" will be used to refer to a compute server or other device acting as an ARP/L-ARP client.

### 1.2. Approach

ARP is a nearly ubiquitous protocol; every device with an Ethernet interface, from hand-helds to hosts, have an implementation of ARP. ARP is plug-and-play; ARP clients do not need configuration to use ARP. That suggests that ARP may be a good fit for devices that want to source and sink MPLS tunnels, but do so in a zero-config, plug-and-play manner, with minimal impact to their code.

The approach taken here is to create a minor variant of the ARP protocol, labeled ARP (L-ARP), which is distinguished by a new hardware type, MPLS-over-Ethernet. Regular (Ethernet) ARP (E-ARP) and L-ARP can coexist; a device, as an ARP client, can choose to send out an E-ARP or an L-ARP request, depending on whether it needs Ethernet or MPLS connectivity. Another device may choose to function as an E-ARP server and/or an L-ARP server, depending on its ability to provide an IP-to-Ethernet and/or IP-to-MPLS mapping.

## 2. Overview of Ethernet ARP

In the most straightforward mode of operation [RFC0826], ARP queries are sent to resolve "directly connected" IP addresses. The ARP request is broadcast, with the Target Protocol Address field (see Section 5 for a description of the fields in an ARP message) carrying the IP address of another node in the same subnet. All the nodes in the LAN receive this ARP request. All the nodes, except the node that owns the IP address, ignore the ARP request. The IP address owner learns the MAC address of the sender from the Source Hardware Address field in the ARP request, and unicasts an ARP reply to the sender. The ARP reply carries the replying node's MAC address in the Source Hardware Address field, thus enabling two-way communication between the two nodes.

A variation of this scheme, known as "proxy ARP" [RFC2002], allows a node to respond to an ARP request with its own MAC address, even when the responding node does not own the requested IP address. Generally, the proxy ARP response is generated by routers to attract traffic for prefixes they can forward packets to. This scheme requires the host to send ARP queries for the IP address the host is trying to reach, rather than the IP address of the router. When there is more than one router connected to a network, proxy ARP enables a host to automatically select an exit router without running any routing protocol to determine IP reachability. Unlike regular ARP, a proxy ARP request can elicit multiple responses, e.g., when more than one router has connectivity to the address being resolved. The sender must be prepared to select one of the responding routers.

Yet another variation of the ARP protocol, called 'Gratuitous ARP' [RFC2002], allows a node to update the ARP cache of other nodes in an unsolicited fashion. Gratuitous ARP is sent as either an ARP request or an ARP reply. In either case, the Source Protocol Address and Target Protocol Address contain the sender's address, and the Source Hardware Address is set to the sender's hardware address. In case of a gratuitous ARP reply, the Target Hardware Address is also set to the sender's address.

### 3. L-ARP Protocol Operation

The L-ARP protocol builds on the proxy ARP model, and also leverages gratuitous ARP model for asynchronous updates.

In this memo, we will refer to L-ARP clients (that make L-ARP requests) and L-ARP servers (that send L-ARP responses). In Figure 1, H1, H2 and H3 are L-ARP clients, and T1, T2 and T3 are L-ARP servers. T4 is a member of the MPLS Fabric that may not be an L-ARP server. Within the MPLS Fabric, the usual MPLS protocols (IGP (i.e., SPRING-MPLS), LDP, RSVP-TE) are run. Say H1, H2 and H3 want to establish MPLS tunnels to each other (for example, they are using BGP MPLS VPNs as the overlay virtual network technology). H1 might also want to talk to a member of the MPLS Fabric, say T. Also, the "protocol" addresses in L-ARP requests are either IPv4 or IPv6 addresses; note that while it is common to use Neighbor Discovery (ND) [RFC4861] for "regular" ARP requests when dealing with IPv6 (i.e., to obtain Ethernet MAC addresses corresponding to an IPv6 address), ND is not used when the ARP request is for an MPLS label.

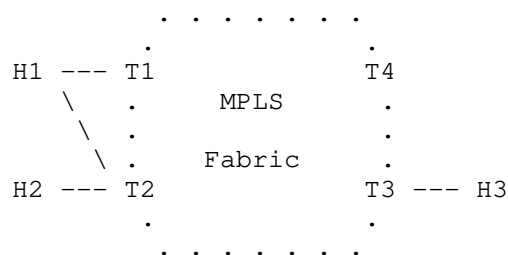


Figure 1: MPLS Fabric

### 3.1. Setup

In Figure 1, the nodes T1-T4, and those in between making up the "MPLS Fabric" are assumed to be running some protocol whereby they can signal MPLS reachability to themselves and to other nodes (like hosts H1-H3). T1-T3 are L-ARP servers; T4 need not be, since it doesn't have an attached L-ARP client. H1-H3 are L-ARP clients.

### 3.2. Egress Operation

A node (say T3) that wants an attached node (say H3) to have MPLS reachability allocates a label L3 to reach H3 and advertises this label into the MPLS Fabric. This can be triggered by configuration on T3, or when T3 first receives an L-ARP request from H3 (indicating that H3 wants MPLS connectivity), or via some other protocol. T3 then advertises (H3, L3) to its peers in the MPLS Fabric so that all members of the Fabric have connectivity to H3. This advertisement can be one of the following:

- \* a "proxy" LDP message (sent on behalf of H3) with prefix H3 and label L3; or
- \* a node Segment ID (SID) advertised on behalf of H3; or
- \* a labeled BGP advertisement, with prefix H3, label L3 and next hop self.

On receiving a packet with label L3, T3 pops the label and send the packet to H3. (In the case of labeled BGP, there would be a two-label stack, with outer label to reach T3 and inner label of L3.) This is the usual operation of an MPLS Fabric, with the addition of advertising labels for nodes outside the fabric.

### 3.3. Ingress Operation

A node (say H1, an L-ARP client) that needs an MPLS tunnel to another node (say H3) identified by a host address (either IPv4 or IPv6) broadcasts over all its interfaces an L-ARP request with the Target Protocol Address set to H3 and Hardware Type set to "MPLS-over-Ethernet". A node receiving the L-ARP request (say T1, an L-ARP server) does the following:

1. checks if it has MPLS reachability to H3. If not, it ignores the L-ARP request.
2. if it does, T1 allocates a label TL3 to reach H3 (if it doesn't already have such a label) and installs an L-FIB entry to swap L1 with the label (stack) to reach H3.
3. sends a (proxy) L-ARP reply to H1 with the Source Hardware Address (SHA) set to (L, M), where M is T1's metric to H3. T1 may also set some attribute bits in the SHA.

### 3.4. Data Plane

To send a packet to H3 over an MPLS tunnel, H1 pushes L1 onto the packet, sets the destination MAC address to M1 and sends it to T1. On receiving this packet, T1 swaps the top label with the label(s) for its MPLS tunnel to H3. If T1's reachability to H3 is via a SPRING label stack, the label L1 acts as an implicit binding SID.

If H1 and H3 have an overlay connection (say an IPVPN [RFC4364] VPN-foo) whereby VM1 on H1 wishes to talk to VM3 on H3 over VPN-foo, H1 does the following:

1. H1 learns information about VPN-foo via BGP (or an SDN controller), including the VPN label VL3 to use to talk to VM3;
2. H1 installs a VRF for VPN-foo, with prefix VM3, label VL3 and next hop H3;
3. H1 binds the local "veth" interface to VM1 to this VRF.
4. When VM1 sends a packet to dest IP address VM3 over its veth interface, H1 looks up VM3 in the corresponding VRF, gets label VL3. It then sends an L-ARP request for next hop H3, and gets TL3.
5. Finally, H1 pushes the label pair (TL3, VL3) onto the packet from VM1 and sends this to T1. This packet will then end up at VM3 on H3.



Note that H1 broadcasts its L-ARP request over its attached interfaces. H1 may receive several L-ARP replies; in that case, H1 can select any subset of these to send MPLS packets destined to H3. As described later, the L-ARP response may contain certain parameters that enable the client to make an informed choice. If the target H3 belongs to one of the subnets that H1 participates in, and H3 is capable of sending L-ARP replies, H1 can use H3's response to send MPLS packets to H3.

#### 4. Attributes

In addition to carrying a label stack to be used in the data plane, an L-ARP reply carries some attributes that are typically used in the control plane. One of these is a metric. The metric is the distance from the L-ARP server to the destination. This allows an L-ARP client that receives multiple responses to decide which ones to use, and whether to load-balance across some of them. The metric typically will be the IGP shortest path distance from server to the destination; this makes comparing metrics from different servers meaningful.

Another attribute is Entropy Label (EL) Capability. This attribute says whether the destination is EL capable (ELC). In Figure 1, if T3 advertises a label to reach H3 and T3 is ELC, T3 can include in its signaling to T1 that it is ELC. In that case, T1's L-ARP reply to H1 can have ELC bit set. This tells H1 that it may include (below the outermost label) an Entropy Label Indicator followed by an Entropy Label. This will help improve load balancing across the MPLS Fabric, and possibly on the last hop to H3.

##### 4.1. Secondary Attributes

Beyond the basic attributes that are carried with every L-ARP request, there more optional attributes, for example, to ask for certain characteristics of the path traffic takes to the destination. These attributes are carried in TLVs that are carried in L-ARP requests and replies.

One such TLV is the Classful Transport (CT: see [I-D.kaliraj-idr-bgp-classful-transport-planes]) TLV. This TLV allows the L-ARP client to request a label to a destination over a tunnel of the given Transport Class. To satisfy this request, the L-ARP server creates (or finds) a tunnel to the destination that is routed over the CT Transport Plane, allocates a label L, inserts an entry in the LFIB to swap L to the tunnel, and sends L to the L-ARP client in its reply.

## 5. L-ARP Message Format

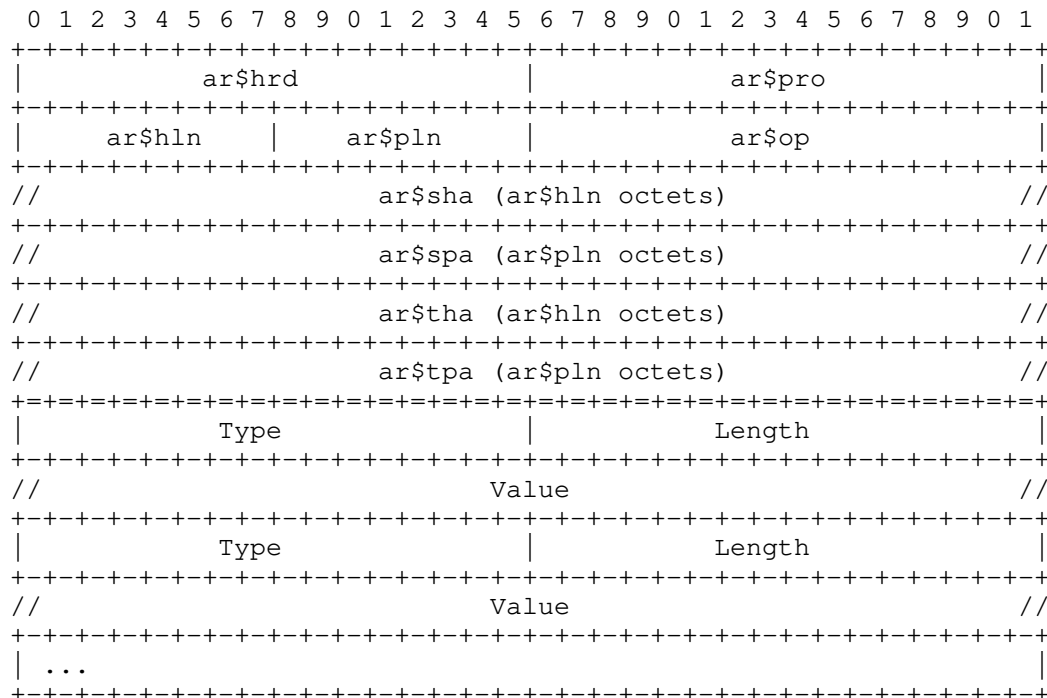


Figure 2: L-ARP Packet Format

ar\$hrd: Hardware Type: MPLS-over-Ethernet. The value of the field used here is HTYPE-MPLS. To start with, we will use the experimental value HW\_EXP2 (256).

ar\$pro: Protocol Type: IPv4/IPv6. The value of the field used here is 0x0800 to resolve an IPv4 address and 0x86DD to resolve an IPv6 address.

ar\$hln: Hardware Address Length: 6

ar\$pln: Protocol Address Length: for an IPv4 address, the length is 4 octets; for an IPv6 address, it is 16.

ar\$op: Operation Code: set to 1 for request, 2 for reply, and 10 for ARP-NAK. Other op codes may be used as needed.

ar\$sha: Source Hardware Address: In an L-ARP request, this is usually all zeros. In an L-ARP reply, Source Hardware Address is the label to reach ar\$spa, as specified in Figure 3 below.

ar\$spa: Source Protocol Address: In an L-ARP request, this field carries the sender's IP address. In an L-ARP reply, this field carries the requested IP address (which may not be the sender's IP address).

ar\$tha: Target Hardware Address: In an L-ARP message, this is all zeros.

ar\$tpa: Target Protocol Address: In an L-ARP request, this field carries the IP address for which the client is seeking an MPLS label.

Type: a 2-octet field defining the Type of the TVL

Length: a 2-octet field defining the Length L of the TVL

Value: an L-octet field with the Value of the TLV

#### 5.1. Hardware Address Format

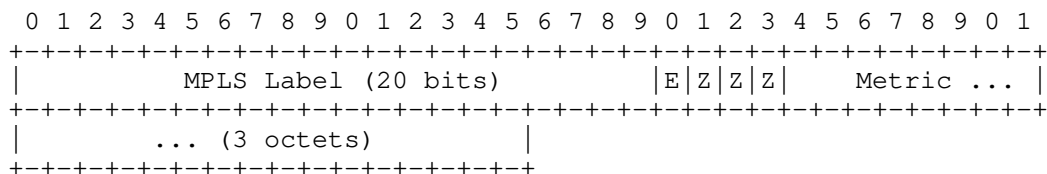


Figure 3: Label Format in L-ARP

MPLS Label: The 20-bit label

E-bit: Entropy Label Capable: this flag indicates whether the corresponding label in the label stack can be followed by an Entropy Label. If this flag is set, the client has the option of inserting ELI and EL as specified in [RFC6790]. The client can choose not to insert ELI/EL pair. If this flag is clear, the client must not insert ELI/EL after the corresponding label.

Z: These bits are not used, and SHOULD be set to zero on sending and ignored on receipt.

Metric: The 3-octet IGP metric to ar\$tha from the point of view of the L-ARP replier.

#### 5.2. CT TLV

The CT TLV has Type (TBD) and Length 4 octets; the Value field consists of the CT attribute.

## 6. Security Considerations

There are many possible attacks on ARP: ARP spoofing, ARP cache poisoning and ARP poison routing, to name a few. These attacks use gratuitous ARP as the underlying mechanism, a mechanism used by L-ARP. Thus, these types of attacks are applicable to L-ARP. Furthermore, ARP does not have built-in security mechanisms; defenses rely on means external to the protocol.

It is well outside the scope of this document to present a general solution to the ARP security problem. One simple answer is to add a TLV that contains a digital signature of the contents of the ARP message. This TLV would be defined for use only in L-ARP messages, although in principle, other ARP messages could use it as well. Such an approach would, of course, need a review and approval by the Security Directorate. If approved, the type of this TLV and its procedures would be defined in this document. If some other technique is suggested, the authors would be happy to include the relevant text in this document, and refer to some other document for the full solution.

## 7. IANA Considerations

IANA is requested to allocate a new ARP hardware type (from registry hrd) for HTYPE-MPLS [IANA].

IANA is further requested to create a registry for Types of L-ARP Secondary Attributes. This registry should contain an entry for the CT Type Section 5.2.

## 8. Acknowledgments

Many thanks to Shane Amante for his detailed comments and suggestions. Many thanks to the team in Juniper prototyping this work for their suggestions on making this variant workable in the context of existing ARP implementations. Thanks too to Luyuan Fang, Alex Semenyaka and Dmitry Afanasiev for their comments and encouragement.

## 9. References

### 9.1. Normative References

- [IANA] IANA, "Address Resolution Protocol (ARP) Parameters/ Hardware Types", n.d., <<https://www.iana.org/assignments/arp-parameters/arp-parameters.xhtml>>.

- [RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <<https://www.rfc-editor.org/info/rfc826>>.
- [RFC2002] Perkins, C., Ed., "IP Mobility Support", RFC 2002, DOI 10.17487/RFC2002, October 1996, <<https://www.rfc-editor.org/info/rfc2002>>.
- [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>>.
- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <<https://www.rfc-editor.org/info/rfc6790>>.
- [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>>.

## 9.2. Informative References

- [I-D.kaliraj-idr-bgp-classful-transport-planes] Vairavakkalai, K., Venkataraman, N., Rajagopalan, B., Mishra, G., Khaddam, M., Xu, X., Szarecki, R. J., and D. J. Gowda, "BGP Classful Transport Planes", Work in Progress, Internet-Draft, draft-kaliraj-idr-bgp-classful-transport-planes-12, 25 August 2021, <<https://www.ietf.org/archive/id/draft-kaliraj-idr-bgp-classful-transport-planes-12.txt>>.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001, <<https://www.rfc-editor.org/info/rfc3107>>.
- [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>>.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <<https://www.rfc-editor.org/info/rfc4364>>.

- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/info/rfc4861>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<https://www.rfc-editor.org/info/rfc5036>>.
- [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.

## Authors' Addresses

Kireeti Kompella  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, 94089  
United States of America

Phone: +1-408-745-2000  
Email: [kireeti.ietf@gmail.com](mailto:kireeti.ietf@gmail.com)

Balaji Rajagopalan  
Juniper Networks  
Survey No.111/1 to 115/4, Wing A & B  
Bangalore 560103  
India

Email: [balajir@juniper.net](mailto:balajir@juniper.net)

Reji Thomas  
Cohesity

Email: [rejithomas.d@gmail.com](mailto:rejithomas.d@gmail.com)

MPLS WG  
Internet-Draft  
Intended status: Standards Track  
Expires: March 11, 2016

K. Kompella  
Juniper Networks, Inc.  
L. Contreras  
Telefonica I+D  
September 8, 2015

Resilient MPLS Rings  
draft-kompella-mpls-rmr-02

Abstract

This document describes the use of the MPLS control and data planes on ring topologies. It describes the special nature of rings, and proceeds to show how MPLS can be effectively used in such topologies. It describes how MPLS rings are configured, auto-discovered and signaled, as well as how the data plane works. Companion documents describe the details of discovery and signaling for specific protocols.

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://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 March 11, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

|                                                |    |
|------------------------------------------------|----|
| 1. Introduction . . . . .                      | 2  |
| 1.1. Definitions . . . . .                     | 3  |
| 2. Motivation . . . . .                        | 4  |
| 3. Theory of Operation . . . . .               | 5  |
| 3.1. Provisioning . . . . .                    | 5  |
| 3.2. Ring Nodes . . . . .                      | 5  |
| 3.3. Ring Links and Directions . . . . .       | 6  |
| 3.3.1. Bypass Links . . . . .                  | 6  |
| 3.4. Ring LSPs . . . . .                       | 7  |
| 3.5. Installing Primary LFIB Entries . . . . . | 7  |
| 3.6. Installing FRR LFIB Entries . . . . .     | 7  |
| 3.7. Protection . . . . .                      | 8  |
| 4. Autodiscovery . . . . .                     | 9  |
| 4.1. Overview . . . . .                        | 9  |
| 4.2. Ring Announcement Phase . . . . .         | 10 |
| 4.3. Mastership Phase . . . . .                | 11 |
| 4.4. Ring Identification Phase . . . . .       | 11 |
| 4.5. Ring Changes . . . . .                    | 11 |
| 5. Ring Signaling . . . . .                    | 12 |
| 6. Ring OAM . . . . .                          | 12 |
| 7. Security Considerations . . . . .           | 12 |
| 8. Acknowledgments . . . . .                   | 12 |
| 9. IANA Considerations . . . . .               | 13 |
| 10. References . . . . .                       | 13 |
| 10.1. Normative References . . . . .           | 13 |
| 10.2. Informative References . . . . .         | 13 |
| Authors' Addresses . . . . .                   | 13 |

## 1. Introduction

Rings are a very common topology in transport networks. A ring is the simplest topology offering link and node resilience. Rings are nearly ubiquitous in access and aggregation networks. As MPLS increases its presence in such networks, and takes on a greater role in transport, it is imperative that MPLS handles rings well; this is not the case today.



This document describes the special nature of rings, and the special needs of MPLS on rings. It then shows how these needs can be met in several ways, some of which involve extensions to protocols such as IS-IS [RFC5305], OSPF[RFC3630], RSVP-TE [RFC3209] and LDP [RFC5036].

The intent of this document is to handle rings that "occur naturally". Many access and aggregation networks in metros have their start as a simple ring. They may then grow into more complex topologies, for example, by adding parallel links to the ring, or by adding "bypass" links. The goal here is to discover these rings (with some guidance), and run MPLS over them efficiently. The intent is not to construct rings in a mesh network, and use those for protection.

### 1.1. Definitions

A (directed) graph  $G = (V, E)$  consists of a set of vertices (or nodes)  $V$  and a set of edges (or links)  $E$ . An edge is an ordered pair of nodes  $(a, b)$ , where  $a$  and  $b$  are in  $V$ . (In this document, the terms node and link will be used instead of vertex and edge.)

A ring is a subgraph of  $G$ . A ring consists of a subset of  $n$  nodes  $\{R_i, 0 \leq i < n\}$  of  $V$ . The directed edges  $\{(R_i, R_{i+1}) \text{ and } (R_{i+1}, R_i), 0 \leq i < n-1\}$  must be a subset of  $E$  (note that index arithmetic is done modulo  $n$ ). We define the direction from node  $R_i$  to  $R_{i+1}$  as "clockwise" (CW) and the reverse direction as "anticlockwise" (AC). As there may be several rings in a graph, we number each ring with a distinct ring ID RID.

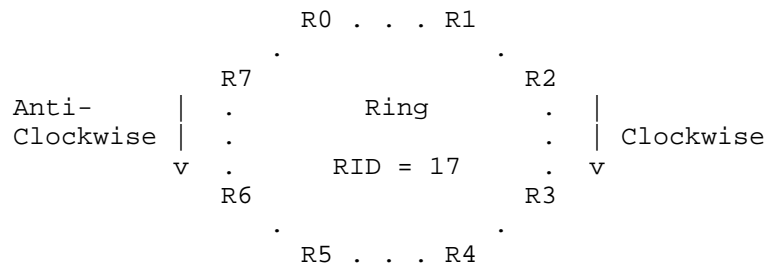


Figure 1: Ring with 8 nodes

The following terminology is used for ring LSPs:

**Ring ID (RID):** A non-zero number that identifies a ring; this is unique in some scope of a Service Provider's network. A node may belong to multiple rings.

Ring node: A member of a ring. Note that a device may belong to several rings.

Node index: A logical numbering of nodes in a ring, from zero upto one less than the ring size. Used purely for exposition in this document.

Ring master: The ring master initiates the ring identification process. Mastership is indicated in the IGP by a two-bit field.

Ring neighbors: Nodes whose indices differ by one (modulo ring size).

Ring links: Links that connect ring neighbors.

Bypass links: Links that connect non-neighboring ring nodes.

Ring direction: A two-bit field in the IGP indicating the direction of a link. The choices are:

UN: 00 undefined link

CW: 01 clockwise ring link

AC: 10 anticlockwise ring link

BY: 11 bypass link

Ring Identification: The process of discovering ring nodes, ring links, link directions, and bypass links.

The following notation is used for ring LSPs:

R<sub>k</sub>: A ring node with index k. R<sub>k</sub> has AC neighbor R<sub>(k-1)</sub> and CW neighbor R<sub>(k+1)</sub>.

RL<sub>k</sub>: A (unicast) Ring LSP anchored on node R<sub>k</sub>.

CL<sub>jk</sub> (AL<sub>jk</sub>): A label allocated by R<sub>j</sub> for RL<sub>k</sub> in the CW (AC) direction.

P<sub>jk</sub> (Q<sub>jk</sub>): A Path (Resv) message sent by R<sub>j</sub> for RL<sub>k</sub>.

## 2. Motivation

A ring is the simplest topology that offers resilience. This is perhaps the main reason to lay out fiber in a ring. Thus, effective mechanisms for fast failover on rings are needed. Furthermore, there

are large numbers of rings. Thus, configuration of rings needs to be as simple as possible. Finally, bandwidth management on access rings is very important, as bandwidth is generally quite constrained here.

The goals of this document are to present mechanisms for improved MPLS-based resilience in ring networks (using ideas that are reminiscent of Bidirectional Line Switched Rings), for automatic bring-up of LSPs, better bandwidth management and for auto-hierarchy. These goals can be achieved using extensions to existing IGP and MPLS signaling protocols, using central provisioning, or in other ways.

### 3. Theory of Operation

Say a ring has ring ID RID. The ring is provisioned by choosing one or more ring masters for the ring and assigning them the RID. Other nodes in the ring may also be assigned this RID, or may be configured as "promiscuous". Ring discovery then kicks in. When each ring node knows its CW and AC ring neighbors and its ring links, and all bypass links have been identified, ring identification is complete.

Once ring identification is complete, each node signals one or more ring LSPs RL<sub>i</sub>. RL<sub>i</sub>, anchored on node R<sub>i</sub>, consists of two counter-rotating unicast LSPs that start and end at R<sub>i</sub>. A ring LSP is "multipoint": any node R<sub>j</sub> can use RL<sub>i</sub> to send traffic to R<sub>i</sub>; this can be in either the CW or AC directions, or both (i.e., load balanced). Both of these counter-rotating LSPs are "active"; the choice of direction to send traffic to R<sub>i</sub> is determined by policy at the node where traffic is injected into the ring. The default is to send traffic along the shortest path. Bidirectional connectivity between nodes R<sub>i</sub> and R<sub>j</sub> is achieved by using two different ring LSPs: R<sub>i</sub> uses RL<sub>j</sub> to reach R<sub>j</sub>, and R<sub>j</sub> uses RL<sub>i</sub> to reach R<sub>i</sub>.

#### 3.1. Provisioning

The goal here is to provision rings with the absolute minimum configuration. The exposition below aims to achieve that using auto-discovery via a link-state IGP (see Section 4). Of course, auto-discovery can be overridden by configuration. For example, a link that would otherwise be classified by auto-discovery as a ring link might be configured not to be used for ring LSPs.

#### 3.2. Ring Nodes

Ring nodes have a loopback address, and run a link-state IGP and an MPLS signaling protocol. To provision a node as a ring node for ring RID, the node is simply assigned that RID. A node may be part of several rings, and thus may be assigned several ring IDs.

To simplify ring provisioning even further, a node N may be made "promiscuous" by being assigned an RID of 0. A promiscuous node listens to RIDs in its IGP neighbors' link-state updates. If N hears a non-zero RID from a neighbor, it joins that ring by taking on that RID. However, if N hears more than one non-zero RID from its neighbors, N remains in promiscuous mode. In many situations, the use of promiscuous mode means that only one or two nodes in the ring needs to be provisioned; everything else is auto-discovered.

A ring node indicates in its IGP updates the ring LSP signaling protocols it supports. This can be LDP and/or RSVP-TE. Ideally, each node should support both.

### 3.3. Ring Links and Directions

Ring links must be MPLS-capable. They are by default unnumbered, point-to-point (from the IGP point of view) and "auto-bundled". The last attribute means that parallel links between ring neighbors are considered as a single link, without the need for explicit configuration for bundling (such as a Link Aggregation Group). Note that each component may be advertised separately in the IGP; however, signaling messages and labels across one component link apply to all components. Parallel links between a pair of ring nodes is often the result of having multiple lambdas or fibers between those nodes. RMR is primarily intended for operation at the packet layer; however, parallel links at the lambda or fiber layer result in parallel links at the packet layer.

A ring link is not provisioned as belonging to the ring; it is discovered to belong to ring RID if both its adjacent nodes belong to RID. A ring link's direction (CW or AC) is also discovered; this process is initiated by the ring's ring master. Note that the above two attributes can be overridden by provisioning if needed; it is then up to the provisioning system to maintain consistency across the ring.

#### 3.3.1. Bypass Links

Bypass links are discovered once ring nodes, ring links and directions have been established. As defined earlier, bypass links are links joining non-neighbor ring nodes; often, this may be the result of optically bypassing ring nodes. The use of bypass links will be described in a future version of this document.

### 3.4. Ring LSPs

Ring LSPs are not provisioned. Once a ring node  $R_i$  knows its RID, its ring links and directions, it kicks off ring LSP signaling automatically.  $R_i$  allocates CW and AC labels for each ring LSP  $RL_k$ .  $R_i$  also initiates the creation of  $RL_i$ . As the signaling propagates around the ring, CW and AC labels are exchanged. When  $R_i$  receives CW and AC labels for  $RL_k$  from its ring neighbors, primary and fast reroute (FRR) paths for  $RL_k$  are installed at  $R_i$ . More details are given in Section 5.

For RSVP-TE LSPs, bandwidths may be signaled in both directions. However, these are not provisioned either; rather, one does "reverse call admission control". When a service needs to use an LSP, the ring node where the traffic enters the ring attempts to increase the bandwidth on the LSP to the egress. If successful, the service is admitted to the ring.

### 3.5. Installing Primary LFIB Entries

In setting up  $RL_k$ , a node  $R_j$  sends out two labels:  $CL_{jk}$  to  $R_{j-1}$  and  $AL_{jk}$  to  $R_{j+1}$ .  $R_j$  also receives two labels:  $CL_{j+1,k}$  from  $R_{j+1}$ , and  $AL_{j-1,k}$  from  $R_{j-1}$ .  $R_j$  can now set up the forwarding entries for  $RL_k$ . In the CW direction,  $R_j$  swaps incoming label  $CL_{jk}$  with  $CL_{j+1,k}$  with next hop  $R_{j+1}$ ; these allow  $R_j$  to act as LSR for  $RL_k$ .  $R_j$  also installs an LFIB entry to push  $CL_{j+1,k}$  with next hop  $R_{j+1}$  to act as ingress for  $RL_k$ . Similarly, in the AC direction,  $R_j$  swaps incoming label  $AL_{jk}$  with  $AL_{j-1,k}$  with next hop  $R_{j-1}$  (as LSR), and an entry to push  $AL_{j-1,k}$  with next hop  $R_{j-1}$  (as ingress).

Clearly,  $R_k$  does not act as ingress for its own LSPs. However, if these LSPs use UHP, then  $R_k$  installs LFIB entries to pop  $CL_{k,k}$  for packets received from  $R_{k-1}$  and to pop  $AL_{k,k}$  for packets received from  $R_{k+1}$ .

### 3.6. Installing FRR LFIB Entries

At the same time that  $R_j$  sets up its primary CW and AC LFIB entries, it can also set up the protection forwarding entries for  $RL_k$ . In the CW direction,  $R_j$  sets up an FRR LFIB entry to swap incoming label  $CL_{jk}$  with  $AL_{j-1,k}$  with next hop  $R_{j-1}$ . In the AC direction,  $R_j$  sets up an FRR LFIB entry to swap incoming label  $AL_{jk}$  with  $CL_{j+1,k}$  with next hop  $R_{j+1}$ . Again,  $R_k$  does not install FRR LFIB entries in this manner.

### 3.7. Protection

In this scheme, there are no protection LSPs as such -- no node or link bypasses, no standby LSPs, no detours, and no LFA-type protection. Protection is via the "other" direction around the ring, which is why ring LSPs are in counter-rotating pairs. Protection works in the same way for link, node and ring LSP failures.

If a node  $R_j$  detects a failure from  $R_{j+1}$  -- either all links to  $R_{j+1}$  fail, or  $R_{j+1}$  itself fails,  $R_j$  switches traffic on all CW ring LSPs to the AC direction using the FRR LFIB entries. If the failure is specific to a single ring LSP,  $R_j$  switches traffic just for that LSP. In either case, this switchover can be very fast, as the FRR LFIB entries can be preprogrammed. Fast detection and fast switchover lead to minimal traffic loss.

$R_j$  then sends an indication to  $R_{j-1}$  that the CW direction is not working, so that  $R_{j-1}$  can similarly switch traffic to the AC direction. For RSVP-TE, this indication can be a PathErr or a Notify; other signaling protocols have similar indications. These indications propagate AC until each traffic source on the ring AC of the failure uses the AC direction. Thus, within a short period, traffic will be flowing in the optimal path, given that there is a failure on the ring. This contrasts with (say) bypass protection, where until the ingress recomputes a new path, traffic will be suboptimal.

Note that the failure of a node or a link will not necessarily affect all ring LSPs. Thus, it is important to identify the affected LSPs (and switch them), but to leave the rest alone.

One point to note is that when a ring node, say  $R_j$ , fails,  $RL_j$  is clearly unusable. However, the above protection scheme will cause a traffic loop:  $R_{j-1}$  detects a failure CW, and protects by sending CW traffic on  $RL_j$  back all the way to  $R_{j+1}$ , which in turn sends traffic to  $R_{j-1}$ , etc. There are three proposals to avoid this:

1. Each ring node acting as ingress sends traffic with a TTL of at most  $2*n$ , where  $n$  is the number of nodes in the ring.
2. A ring node sends protected traffic (i.e., traffic switched from CW to AC or vice versa) with TTL just large enough to reach the egress.
3. A ring node sends protected traffic with a special purpose label below the ring LSP label. A protecting node first checks for the presence of this label; if present, it means that the traffic is looping and MUST be dropped.

It is recommended that (2) be implemented. The other methods are optional.

#### 4. Autodiscovery

##### 4.1. Overview

Auto-discovery proceeds in three phases. The first phase is the announcement phase. The second phase is the mastership phase. The third phase is the ring identification phase.

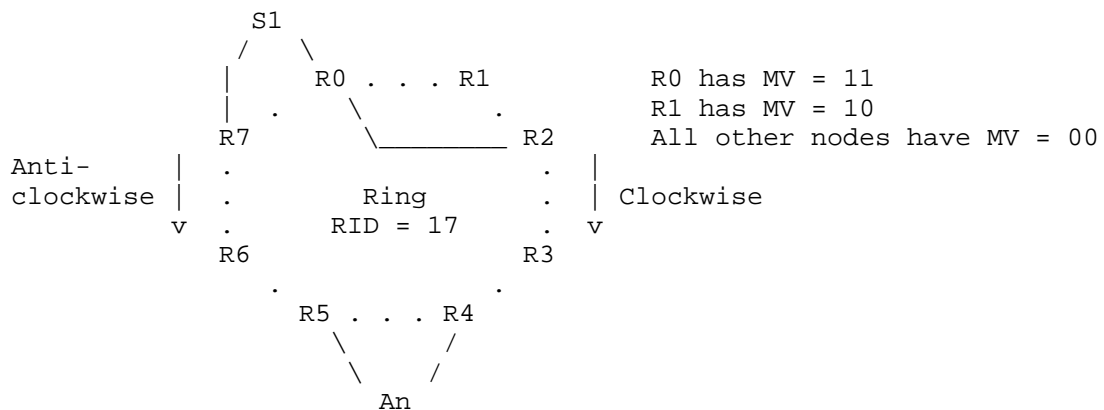
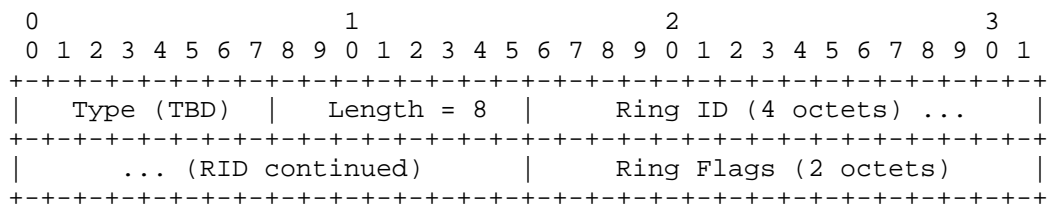


Figure 2: Ring with non-ring nodes and links

In what follows, we refer to a ring Type-Length-Value (TLV). This is a new TLV that contains an RID and associated flags. A ring link TLV is a ring TLV that appears as a sub-TLV of a traffic engineering TLV (TE TLV) of each link that is identified as a ring link or a bypass link. For IS-IS, the TE TLV is the extended reachability TLV; for OSPF, it is the Link TLV in the opaque TE LSA. A ring node TLV is a ring TLV that appears as a sub-TLV of a "node TLV" once for each ring this node is participating in. In IS-IS, the node TLV is the Router ID TLV; in OSPF, it is a new top-level TLV of the TE LSA. The ring direction field is ignored in ring node TLVs.



IS-IS Ring TLV Format

```

      0                               1                               2                               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Type (TBD)                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Ring ID (4 octets)                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Ring Flags (2 octets)      |      Pad (2 octets)      |
+-----+-----+-----+-----+-----+-----+-----+-----+
Pad is set to zero when sending and ignored on receipt.

```

#### OSPF Ring TLV Format

```

      0                               1
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-----+-----+-----+-----+-----+-----+-----+-----+
|MV|RD|SP|OP|M|      MBZ      |
+-----+-----+-----+-----+-----+-----+-----+
MV: Mastership Value
RD: Ring Direction
SP: Signaling Protocols (10 = RSVP-TE; 01 = LDP)
OP: OAM Protocols (10 = BFD; 01 = EFM)
M : Elected Master (0 = no, 1 = yes)

```

#### Ring Flags Format

### 4.2. Ring Announcement Phase

Each node participating in an MPLS ring is assigned an RID; in the example, RID = 17. A node is also provisioned with a mastership value. Each node advertises a ring node TLV for each ring it is participating in, along with the associated flags. It then starts timer T1.

A node in promiscuous mode doesn't advertise any ring node TLVs. If it hears exactly one non-zero RID from its IGP neighbors, it joins that ring, and sends one ring node TLV with that RID. If it hears more than one RID from its IGP neighbors, it doesn't join any rings, and withdraws any ring node TLVs it may have advertised.

The announcement phase allows a ring node to discover other ring nodes in the same ring so that a ring master can be elected and ring links be identified.



#### 4.3. Mastership Phase

When timer T1 fires, a node enters the mastership phase. In this phase, each ring node N starts timer T2 and checks if it is master. If it is the node with the lowest loopback address of all nodes with the highest mastership values, N declares itself master by readvertising its ring node TLV with the M bit set.

When timer T2 fires, each node examines the ring node TLVs from all other nodes in the ring to identify the ring master. There should be exactly one; if not, each node restarts timer T2 and tries again. The nodes that set their M bit should be extra careful in advertising their M bit in subsequent tries.

#### 4.4. Ring Identification Phase

When there is exactly one ring master M, M enters the Ring Identification Phase. M indicates that it has successfully completed this phase by advertising ring link TLVs. This is the trigger for M's CW neighbor to enter the Ring Identification Phase. This phase passes CW until all ring nodes have completed ring identification.

In the Ring Identification Phase, a node X that has two or more IGP neighbors that belong to the ring picks one of them to be its CW ring neighbor. If X is the ring master, it also picks a node as its AC ring neighbor. If there are exactly two such nodes, this step is trivial. If not, X computes a ring that includes all nodes that have completed the Ring Identification Phase (as seen by their ring link TLVs) and further contains the maximal number of nodes that belong to the ring. Based on that, X picks a CW neighbor and inserts ring link TLVs with ring direction CW for each link to its CW neighbor; X also inserts a ring link TLV with direction AC for each link to its AC neighbor. Then, X determines its bypass links. These are links connected to ring nodes that are not ring neighbors. X advertises ring link TLVs for bypass links by setting the link direction to "bypass link".

#### 4.5. Ring Changes

The main changes to a ring are:

- ring link addition;
- ring link deletion;
- ring node addition; and
- ring node deletion.

The main goal of handling ring changes is (as much as possible) not to perturb existing ring operation. Thus, if the ring master hasn't changed, all of the above changes should be local to the point of change. Link adds just update the IGP; signaling should take advantage of the new capacity as soon as it learns. Link deletions in the case of parallel links also show up as a change in capacity (until the last link in the bundle is removed.)

The removal of the last ring link between two nodes, or the removal of a ring node is an event that triggers protection switching. In a simple ring, the result is a broken ring. However, if a ring has bypass links, then it may be able to converge to a smaller ring with protection. Details of this process will be given in a future version.

The addition of a new ring node can also be handled incrementally. Again, the details of this process will be given in a future version.

## 5. Ring Signaling

A future version of this document will specify protocol-independent details about ring LSP signaling.

## 6. Ring OAM

Each ring node should advertise in its ring node TLV the OAM protocols it supports. Each ring node is expected to run a link-level OAM over each ring and bypass link. This should be an OAM protocol that both neighbors agree on. The default hello time is 3.3 millisecond.

Each ring node also sends OAM messages over each direction of its ring LSP. This is a multi-hop OAM to check LSP liveness; typically, BFD would be used for this. The node chooses the hello interval; the default is once a second.

## 7. Security Considerations

It is not anticipated that either the notion of MPLS rings or the extensions to various protocols to support them will cause new security loopholes. As this document is updated, this section will also be updated.

## 8. Acknowledgments

Many thanks to Pierre Bichon whose exemplar of self-organizing networks and whose urging for ever simpler provisioning led to the notion of promiscuous nodes.

## 9. IANA Considerations

There are no requests as yet to IANA for this document.

## 10. References

### 10.1. 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, <<http://www.rfc-editor.org/info/rfc2119>>.

### 10.2. Informative References

- [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, <<http://www.rfc-editor.org/info/rfc3209>>.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <<http://www.rfc-editor.org/info/rfc3630>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<http://www.rfc-editor.org/info/rfc5036>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<http://www.rfc-editor.org/info/rfc5305>>.

## Authors' Addresses

Kireeti Kompella  
Juniper Networks, Inc.  
1194 N. Mathilda Avenue  
Sunnyvale, CA 94089  
USA

Email: [kireeti.kompella@gmail.com](mailto:kireeti.kompella@gmail.com)

Luis M. Contreras  
Telefonica I+D  
Ronda de la Comunicacion  
Sur-3 building, 3rd floor  
Madrid 28050  
Spain

Email: [luismiguel.contrerasmurillo@telefonica.com](mailto:luismiguel.contrerasmurillo@telefonica.com)  
URI: <http://people.tid.es/LuisM.Contreras/>

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: April 19, 2016

Z. Li  
Q. Zhao  
Huawei Technologies  
T. Yang  
China Mobile  
R. Raszuk  
Individual  
L. Fang  
Microsoft  
October 17, 2015

Usecases of MPLS Global Label  
draft-li-mpls-global-label-usecases-03

Abstract

As the MPLS technologies develop, MPLS label is not only used with the local meaning which is always be understood by the upstream node and the downstream node, but also used with the global meaning which can be understood by all nodes or part of nodes in the network. The document defines the latter as the global label and proposes the possible use cases of global label. In these usecases MPLS global label can be used for location identification, VPN identification, segment routing, etc.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 19, 2016.

## Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

|                                                           |    |
|-----------------------------------------------------------|----|
| 1. Introduction . . . . .                                 | 2  |
| 2. Terminology . . . . .                                  | 3  |
| 3. Use Cases . . . . .                                    | 3  |
| 3.1. Location Identification . . . . .                    | 3  |
| 3.2. VPN Identification . . . . .                         | 4  |
| 3.2.1. Flow Label of VPN LSP . . . . .                    | 4  |
| 3.2.2. Aggregate MVPN/VPLS over Single P-Tunnel . . . . . | 5  |
| 3.3. Segment Routing . . . . .                            | 5  |
| 4. Discussion . . . . .                                   | 6  |
| 5. IANA Considerations . . . . .                          | 8  |
| 6. Security Considerations . . . . .                      | 8  |
| 7. References . . . . .                                   | 8  |
| 7.1. Normative References . . . . .                       | 8  |
| 7.2. Informative References . . . . .                     | 8  |
| Authors' Addresses . . . . .                              | 11 |

## 1. Introduction

In the traditional MPLS architecture, MPLS label is always distributed from the downstream node to the upstream node by LDP, RSVP-TE and MP-BGP. These label mappings always have the local meaning which can only be understood by the upstream node and the downstream node. As the MPLS technologies develop, there proposes possible usecases in which MPLS label mapping can be advertised to all nodes or part of nodes in the network. That is, the meaning of the label mapping will be understood by all nodes or part of nodes in the network other than the local upstream node and downstream node. This document defines such type of MPLS label as global label as the opposite of local label.

In the MPLS world there are another pair of label related concepts: per-platform label space [RFC3031] and context-specific label space [RFC5331]. According to [RFC3031] MPLS local label can be allocated from per-platform label space and per-interface label space (in [RFC5331], per-interface label space is generalized as one type of context-specific label space). MPLS global label can also be allocated from per-platform label space or context-specific label space.

The document proposes the possible usecases of MPLS global label. In these usecases MPLS global label can be used for location identification, VPN identification, segment routing, etc.

## 2. Terminology

CE: Customer Edge

MP2P: Multi-Point to Point

MP2MP: Multi-point to Multi-point

MVPN: Multicast VPN

P2MP: Point to Multi-Point

P2P: Point to Point

PE: Provider Edge

## 3. Use Cases

### 3.1. Location Identification

[I-D.bryant-mpls-flow-ident] and [I-D.bryant-mpls-synonymous-flow-labels] propose the challenge of the measurement of packet loss for the multi-point to point LSP. In this case the same label is normally used by multiple ingress or upstream LSRs for specific prefixes and hence source identification is not possible by inspection of the top label by the egress LSRs. Thus [I-D.bryant-mpls-synonymous-flow-labels] proposes the synonymous flow label to be used to introduce some source specific information encapsulated in the packet to identify packet batches from a specific source.

MPLS LDP LSP is one type of multi-point to point LSP. As the network convergence develops, MPLS LDP network needs to interwork with MPLS TE/MPLS-TP network and unified MPLS OAM becomes the realistic requirement. In this usecase, MPLS global label can be allocated for

each network node and advertised in the network. When implement the measurement of packet loss for LDP LSP, such MPLS global label can be used as the flow label to identify the source node of the LDP LSP. When the destination receives the packets it can differentiate flows from specific source node based on the advertised global label binding information for network nodes. In this usecase, MPLS global label is used as the unique identification of source nodes in the network and may save the complex flow label negotiation process between the source node and the destination node.

### 3.2. VPN Identification

MPLS global label can be allocated for VPN and advertised in the network. In this usecase, MPLS global label is used as the unique identification of VPN in the network and can be used for multiple purposes.

#### 3.2.1. Flow Label of VPN LSP

BGP VPN LSP is another type of multi-point to point LSP which faces the challenge of the measurement of packet loss proposed by [I-D.bryant-mpls-flow-ident] and [I-D.bryant-mpls-synonymous-flow-labels]. In this usecase, the flow label should be introduced to identification of the source VPN. There are two possible ways to use global label as the flow label:

Option 1: The global label is allocated for the same VPN on all PE nodes and advertised in the network. And global labels can be allocated for PE nodes and advertised in the network. Then the flow label should be the source PE label + the VPN label shown in the figure 1.

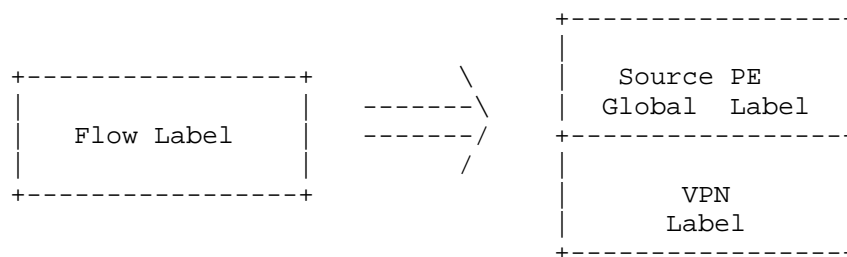


Figure 1: Flow Label using Two Layers of Global Label

Option 2: The global label is allocated directly for source VPN (identified by the pair of { Source PE, VPN }) and advertised in the network. We call such label as Source VPN label. The flow label should be the source VPN label shown in the figure 2.



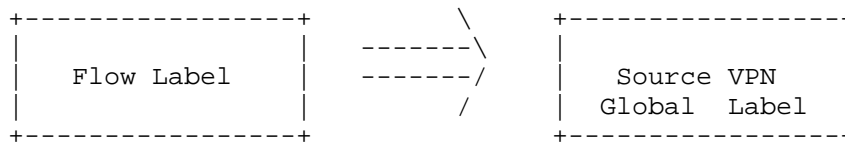


Figure 2: Flow Label using One Layer of Global Label

No matter option 1 or option 2 is adopted, when the destination receives the packets it can differentiate flows from specific source VPN based on the advertised global label binding information.

### 3.2.2. Aggregate MVPN/VPLS over Single P-Tunnel

In BGP-base Multicast VPN ([RFC6513]) and VPLS Multicast([RFC7117]), in order to implement aggregating multiple MVPN/VPLS Instances on a single P-Tunnel (i.e. sharing one P2MP LSP) , context-specific label is introduced to identify the MVPN/VPLS instance and the label binding is allocated by the root PE and advertised to the leaf PEs. In this usecase the context-specific label is one type of global label to uniquely identify the MVPN/VPLS instance in the network.

The context-specific label can solve the issue of aggregating multiple MVPNs or VPLS instances over a single P2MP LSP. But if the MP2MP LSP is adopted for aggregating multiple MVPN/VPLS instances the solution does not work since there are multiple root PEs which may allocate the same context-specific label for different MVPN/VPLS instances. In order to solve the issue the global label can be allocated to the same MVPN/VPLS instance on all PEs and advertised in the network. Then the global label will become the unique identification of VPN instance in the network. When aggregating multiple MVPNs or VPLS instances over one MP2MP LSP, the corresponding MPLS global label binding with the MVPN/VPLS instance can be encapsulated by the root PE. Then the leaf PEs can determine the MVPN or VPLS instance the received packets belong to based on the advertised global label binding information for MVPN/VPLS instances. The solution can provide the unified solution for aggregating multiple MVPN/VPLS instances over P2MP LSP and MP2MP LSP. And the solution can save the complex control plane and forwarding plane process of context-specific label.

### 3.3. Segment Routing

Segment Routing [I-D.ietf-spring-segment-routing] is introduced to leverage the source routing paradigm for traffic engineering, fast re-route, etc. A node can steer a packet through an ordered list of segments. A segment can represent any instruction, topological or

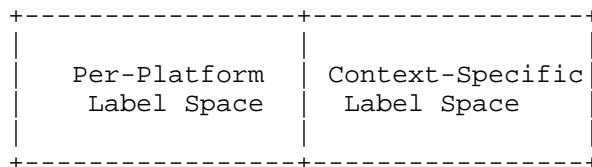
service-based. Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane in which a segment can be encoded as an MPLS label and an ordered list of segments can be encoded as a stack of labels.

Segment Routing [I-D.ietf-spring-segment-routing] introduces some segments such as node segment, adjacency segment, etc. SR Global Block (SRGB) is also introduced for allocation of segment. In the MPLS architecture, SRGB is the set of local labels reserved for global segments. When the global segment index is advertised, it can be transitioned to MPLS label based on the SRGB. According to [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-isis-segment-routing-extensions] MPLS global label binding information can also be directly advertised in the network. For example, in the section 2.1 of [I-D.ietf-ospf-segment-routing-extensions], when the Length field of SID/Label Sub-TLV is set as 3, it will represent the label which can be flooded in the whole network. By this method MPLS global label can be directly allocated for specific node or adjacency, etc. and advertised in the network. The solution can save the complex process of SRGB advertisement and transition from the global Segment ID to MPLS label.

#### 4. Discussion

In the MPLS world, we can adopt the dichotomy to divide it into per-platform label space and context-specific label space.

##### MPLS World



When we adopt another dichotomy to divide the MPLS world into local label and global label, we may face more challenges.

| Local Label     |            | vs. | Global Label                                                            |            |
|-----------------|------------|-----|-------------------------------------------------------------------------|------------|
|                 |            |     | Special Purpose Label (RFC 7274)                                        |            |
|                 |            |     | MPLS Upstream Label Assignment /Context-Specific Label Space (RFC 5331) |            |
| LDP             | (RFC 5036) |     | Entropy Label                                                           | (RFC 6790) |
| RSVP-TE         | (RFC 3209) |     | Flow Label                                                              | (RFC 6391) |
| BGP LSP         | (RFC 3107) |     |                                                                         |            |
| L3VPN           | (RFC 4364) |     | BGP-base VPLS (RFC 4761)                                                |            |
| LDP-based L2VPN | (RFC 4762) |     | Segment Routing                                                         |            |
| EVPN            | (RFC 7432) |     | (draft-ietf-spring-segment-routing)                                     |            |
|                 |            |     | Domain-Wide Label                                                       |            |
|                 |            |     | (Usecases: Synonymous Label/Segment Routing, etc.)                      |            |

Figure 3: Division of MPLS World Using Local Label and Global Label

In the figure 3, we can easily understand the local label using for LDP, RSVP-TE, label BGP, L3VPN, LDP-based L2VPN, EVPN, etc. But for the opposite of these applications there may be many usecases which are different from each other, but share the common characteristic that the label meaning can be understood by all network nodes or part of network nodes instead of only the local downstream nodes and upstream nodes for which in this document such label is defined as global label :

-- For special purpose labels, their meaning can be understood by all nodes in the MPLS network. Should they belong to global label?

-- For MPLS upstream label assignment in context-specific label space, all downstream nodes can understand the meaning of the label allocated by the upstream node binding for specific MVPN/VPLS instance. We can see the root PE as one type to central controlled node to allocate label to all leaf nodes. And thinking about the uniqueness of the context determine by the shared P-tunnel, these labels in fact are also unique in the network. Should they belong to global label?

-- For entropy label and flow label, the label is calculated by the ingress node based on specific hash algorithms which is totally different from the local label distributed in the MPLS control plane.

And all nodes along the path will parse the label and according to the uniform meaning to use the label for ECMP. But the label values can be duplicate since they are calculated by different ingress nodes. Should they belong to global label?

-- For BGP-based VPLS and Segment Routing, they can adopt the local label block. But they introduce the global ID and transit them into the local label. Especially for segment routing, when all nodes in the network adopts the same SRGB, the global segment ID is easily transited to a unique global label value in the network. Should they belong to global label?

-- This document proposes some usecases to directly allocate the unique label value and advise the label binding in the network. Should they be directly called as global label or Domain-Wide label as one type of global label?

Since above applications which are different from the traditional MPLS local label, can we define all of them as global label or define some of them as global label and bring some use cases to the local label field? Or maybe such dichotomy using local label and global label does not exist.

## 5. IANA Considerations

This document makes no request of IANA.

## 6. Security Considerations

TBD.

## 7. References

### 7.1. 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, <<http://www.rfc-editor.org/info/rfc2119>>.

### 7.2. Informative References

[I-D.bryant-mpls-flow-ident]  
Bryant, S., Pignataro, C., Chen, M., Li, Z., and G. Mirsky, "MPLS Flow Identification", draft-bryant-mpls-flow-ident-02 (work in progress), September 2015.

- [I-D.bryant-mpls-synonymous-flow-labels]  
Bryant, S., Swallow, G., Sivabalan, S., Mirsky, G., Chen, M., and Z. Li, "RFC6374 Synonymous Flow Labels", draft-bryant-mpls-synonymous-flow-labels-01 (work in progress), July 2015.
- [I-D.ietf-isis-segment-routing-extensions]  
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-05 (work in progress), June 2015.
- [I-D.ietf-ospf-segment-routing-extensions]  
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-05 (work in progress), June 2015.
- [I-D.ietf-spring-segment-routing]  
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and r. rjs@rob.sh, "Segment Routing Architecture", draft-ietf-spring-segment-routing-06 (work in progress), October 2015.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <<http://www.rfc-editor.org/info/rfc3031>>.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001, <<http://www.rfc-editor.org/info/rfc3107>>.
- [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, <<http://www.rfc-editor.org/info/rfc3209>>.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <<http://www.rfc-editor.org/info/rfc4364>>.
- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<http://www.rfc-editor.org/info/rfc4761>>.

- [RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <<http://www.rfc-editor.org/info/rfc4762>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<http://www.rfc-editor.org/info/rfc5036>>.
- [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <<http://www.rfc-editor.org/info/rfc5331>>.
- [RFC6391] Bryant, S., Ed., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, DOI 10.17487/RFC6391, November 2011, <<http://www.rfc-editor.org/info/rfc6391>>.
- [RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February 2012, <<http://www.rfc-editor.org/info/rfc6513>>.
- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <<http://www.rfc-editor.org/info/rfc6790>>.
- [RFC7117] Aggarwal, R., Ed., Kamite, Y., Fang, L., Rekhter, Y., and C. Kodeboniya, "Multicast in Virtual Private LAN Service (VPLS)", RFC 7117, DOI 10.17487/RFC7117, February 2014, <<http://www.rfc-editor.org/info/rfc7117>>.
- [RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <<http://www.rfc-editor.org/info/rfc7274>>.
- [RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <<http://www.rfc-editor.org/info/rfc7432>>.

## Authors' Addresses

Zhenbin Li  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: lizhenbin@huawei.com

Quintin Zhao  
Huawei Technologies  
125 Nagog Technology Park  
Acton, MA 01719  
US

Email: quintin.zhao@huawei.com

Tianle Yang  
China Mobile  
32, Xuanwumenxi Ave.  
Beijing 01719  
China

Email: yangtianle@chinamobile.com

Robert Raszuk  
Individual

Email: robert@raszuk.net

Luyuan Fang  
Microsoft  
5600 148th Ave NE  
Redmond, WA 98052  
USA

Email: lufang@microsoft.com

MPLS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 9, 2017

K. Raza  
R. Asati  
Cisco Systems, Inc.

X. Liu  
Ericsson

S. Esale  
Juniper Networks

X. Chen  
Huawei Technologies

H. Shah  
Ciena Corporation

July 8, 2016

YANG Data Model for MPLS LDP and mLDP  
draft-raza-mpls-ldp-mldp-yang-04

Abstract

This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).

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 <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2017.



## Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

|                                               |     |
|-----------------------------------------------|-----|
| 1. Introduction . . . . .                     | 3   |
| 2. Specification of Requirements . . . . .    | 3   |
| 3. LDP YANG Model . . . . .                   | 3   |
| 3.1. Overview . . . . .                       | 4   |
| 3.2. Configuration . . . . .                  | 7   |
| 3.2.1. Configuration Hierarchy . . . . .      | 11  |
| 3.2.2. All-VRFs Configuration . . . . .       | 14  |
| 3.3. Operational State . . . . .              | 14  |
| 3.3.1. Derived States . . . . .               | 21  |
| 3.4. Notifications . . . . .                  | 26  |
| 3.5. Actions . . . . .                        | 26  |
| 4. mLDP YANG Model . . . . .                  | 27  |
| 4.1. Overview . . . . .                       | 27  |
| 4.2. Configuration . . . . .                  | 28  |
| 4.2.1. Configuration Hierarchy . . . . .      | 28  |
| 4.2.2. mldp container . . . . .               | 30  |
| 4.2.3. Leveraging LDP containers . . . . .    | 31  |
| 4.2.4. YANG tree . . . . .                    | 31  |
| 4.3. Operational State . . . . .              | 33  |
| 4.3.1. Derived states . . . . .               | 38  |
| 4.4. Notifications . . . . .                  | 42  |
| 4.5. Actions . . . . .                        | 43  |
| 5. Open Items . . . . .                       | 43  |
| 6. YANG Specification . . . . .               | 43  |
| 7. Security Considerations . . . . .          | 110 |
| 8. IANA Considerations . . . . .              | 110 |
| 9. Acknowledgments . . . . .                  | 110 |
| 10. References . . . . .                      | 110 |
| 10.1. Normative References . . . . .          | 110 |
| 10.2. Informative References . . . . .        | 113 |
| Appendix A. Additional Contributors . . . . . | 113 |

|                    |     |
|--------------------|-----|
| Authors' Addresses | 113 |
|--------------------|-----|

## 1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modelling language for the NETCONF.

This document introduces a YANG data model for MPLS Label Distribution Protocol (LDP) [RFC5036] and Multipoint LDP (mLDP) [RFC6388]. For LDP, it also covers LDP IPv6 [RFC7552] and LDP capabilities [RFC5561].

The data model is defined for following constructs that are used for managing the protocol:

- o Configuration
- o Operational State
- o Executables (Actions)
- o Notifications

This document is organized to define the data model for each of the above constructs (configuration, state, action, and notifications) in the sequence as listed earlier. Given that mLDP is tightly coupled with LDP, mLDP data model is defined under LDP tree and in the same sequence as listed above.

## 2. Specification of Requirements

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

In this document, the word "IP" is used to refer to both IPv4 and IPv6, unless otherwise explicitly stated. For example, "IP address family" means and be read as "IPv4 and/or IPv6 address family"

## 3. LDP YANG Model

### 3.1. Overview

This document defines a new module named "ietf-mpls-ldp" for LDP/mLDP data model where this module augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg].

There are four main containers in "ietf-mpls-ldp" module as follows:

- o Read-Write parameters for configuration (Discussed in Section 3.2)
- o Read-only parameters for operational state (Discussed in Section 3.3)
- o Notifications for events (Discussed in Section 3.4)
- o RPCs for executing commands to perform some action (Discussed in Section 3.5)

For the configuration and state data, this model follows the similar approach described in [I-D.openconfig-netmod-opstate] to represent the configuration (intended state) and operational (applied and derived) state. This means that for every configuration (rw) item, there is an associated (ro) item under "state" container to represent the applied state. Furthermore, protocol derived state is also kept under "state" tree corresponding to the protocol area (discovery, peer etc.). [Ed note: This document will be (re-)aligned with [I-D.openconfig-netmod-opstate] once that specification is adopted as a WG document]

Following diagram depicts high level LDP yang tree organization and hierarchy:

```

module: ietf-mpls-ldp
  +-- rw routing
    +-- rw control-plane-protocols
      +-- rw mpls-ldp
        +-- rw global
          +-- rw config
            +-- rw ...
          +-- ro state
            +-- ro ...
          .
        +-- rw ...
      +-- rw ...
    ...

rpcs:
  +-- x mpls-ldp-rpc
  +-- x . . . . .

notifications:
  +--- n mpls-ldp-notif
  +--- n ...

```

Figure 1

Before going into data model details, it is important to take note of the following points:

- o This module aims to address only the core LDP/mLDP parameters as per RFC specification, as well as some widely used and deployed non-RFC features (such as label policies, session authentication etc). Any vendor specific feature should be defined in a vendor-specific augmentation of this model.
- o Multi-topology LDP [RFC7307] and Multi-topology mLDP [I-D.iwijnand-mpls-mldp-multi-topology] are beyond the scope of this document.
- o This module does not cover any applications running on top of LDP and mLDP, nor does it cover any OAM procedures for LDP and mLDP.
- o This model is a VPN Forwarding and Routing (VRF)-centric model. It is important to note that [RFC4364] defines VRF tables and default forwarding tables as different, however from a yang modelling perspective this introduces unnecessary complications,

hence we are treating the default forwarding table as just another VRF.

- o A "network-instance" as defined in [I-D.rtgyangdt-rtgwg-ni-model] refers to a VRF instance (both default and non-default) within the scope of this model.
- o This model supports two address-families, namely "ipv4" and "ipv6".
- o This model assumes platform-wide label space (i.e. label space Id of zero). However, when Upstream Label assignment [RFC6389] is in use, an upstream assigned label is looked up in a Context-Specific label space as defined in [RFC5331].
- o The label and peer policies (including filters) are defined using a prefix-list. When used for a peer policy, the prefix refers to the LSR Id of the peer. The prefix-list is referenced from routing-policy model as defined in [I-D.ietf-rtgwg-policy-model].
- o The use of grouping (templates) for bundling and grouping the configuration items is not employed in current revision, and is a subject for consideration in future.
- o This model uses the terms LDP "neighbor"/"adjacency", "session", and "peer" with the following semantics:
  - \* Neighbor/Adjacency: An LDP enabled LSR that is discovered through LDP discovery mechanisms.
  - \* Session: An LDP neighbor with whom a TCP connection has been established.
  - \* Peer: An LDP session which has successfully progressed beyond its initialization phase and is either already exchanging the bindings or is ready to do so.

It is to be noted that LDP Graceful Restart mechanisms defined in [RFC3478] allow keeping the exchanged bindings for some time after a session goes down with a peer. We call such a state -- i.e. keeping peer bindings without established or recovered peering -- a "stale" peer. When used in this document, the above terms will refer strictly to the semantics and definitions defined for them.

A graphical representation of LDP YANG data model is presented in Figure 3, Figure 5, Figure 11, and Figure 12. Whereas, the actual model definition in YANG is captured in Section 6.

While presenting the YANG tree view and actual .yang specification, this document assumes the reader is familiar with the concepts of YANG modeling, its presentation and its compilation.

### 3.2. Configuration

This specification defines the configuration parameters for base LDP as specified in [RFC5036] and LDP IPv6 [RFC7552]. Moreover, it incorporates provisions to enable LDP Capabilities [RFC5561], and defines some of the most significant and commonly used capabilities such as Typed Wildcard FEC [RFC5918], End-of-LIB [RFC5919], and LDP Upstream Label Assignment [RFC6389].

This specification supports VRF-centric configuration. For implementations that support protocol-centric configuration, with provision for inheritance and items that apply to all vrfs, we recommend an augmentation of this model such that any protocol-centric or all-vrf configuration is defined under their designated containers within the standard network-instance (please see Section 3.2.2)

This model augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg]. For LDP interfaces, this model refers the MPLS interface as defined under MPLS base specification [I-D.saad-mpls-base-yang]. Furthermore, as mentioned earlier, the configuration tree presents read-write intended configuration leave/items as well as read-only state of the applied configuration. The former is listed under "config" container and latter under "state" container.

Following is high-level configuration organization for LDP/mLDP:

```

module: ietf-mpls-ldp
  +-- routing
  +-- control-plane-protocols
    +-- mpls-ldp
      +-- global
        +-- ...
        +-- ...
        +-- address-family* [afi]
          +-- . . .
          +-- . . .
        +-- discovery
          +-- . . .
      +-- peers
        +-- ...
        +-- ...

```

Figure 2

Given the configuration hierarchy, the model allows inheritance such that an item in a child tree is able to derive value from a similar or related item in one of the parent. For instance, hello holdtime can be configured per-VRF or per-VRF-interface, thus allowing inheritance as well flexibility to override with a different value at any child level.

Following is a simplified graphical representation of the data model for LDP configuration

```

+--rw mpls-ldp!
  +--rw global
    |
    | +--rw config
    | |
    | | +--rw capability
    | | |
    | | | +--rw end-of-lib {capability-end-of-lib}?
    | | | | +--rw enable?    boolean
    | | | +--rw typed-wildcard-fec {capability-typed-wildcard-fec}?
    | | | | +--rw enable?    boolean
    | | | +--rw upstream-label-assignment {capability-upstream-label-assign
ment}?
    | | | | +--rw enable?    boolean
    | | | +--rw graceful-restart
    | | | | +--rw enable?          boolean
    | | | | +--rw helper-enable?   boolean {graceful-restart-helper-mod
e}?
    | | | +--rw reconnect-time?    uint16
    | | | +--rw recovery-time?     uint16
    | | | +--rw forwarding-holdtime? uint16
    | | +--rw igp-synchronization-delay? uint16
    | +--rw lsr-id?                yang:dotted-quad

```

```

|   +--rw address-family* [afi]
|   |   +--rw afi          ldp-address-family
|   |   +--rw config
|   |   |   +--rw enable?          boolean
|   |   |   +--rw label-policy
|   |   |   |   +--rw independent-mode
|   |   |   |   |   +--rw assign {policy-label-assignment-config}?
|   |   |   |   |   |   +--rw (prefix-option)?
|   |   |   |   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   |   |   |   |   |   +--rw host-routes-only?      boolean
|   |   |   |   +--rw advertise
|   |   |   |   |   +--rw explicit-null
|   |   |   |   |   |   +--rw enable?          boolean
|   |   |   |   |   |   +--rw prefix-list?      prefix-list-ref
|   |   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   |   +--rw accept
|   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   +--rw ordered-mode {policy-ordered-label-config}?
|   |   |   +--rw egress-lsr
|   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   |   +--rw advertise
|   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   |   +--rw accept
|   |   |   |   +--rw prefix-list?          prefix-list-ref
|   |   +--rw ipv4
|   |   |   +--rw transport-address?      inet:ipv4-address
|   |   +--rw ipv6
|   |   |   +--rw transport-address?      inet:ipv6-address
|   +--rw discovery
|   |   +--rw interfaces
|   |   |   +--rw config
|   |   |   |   +--rw hello-holdtime?      uint16
|   |   |   |   +--rw hello-interval?      uint16
|   |   |   +--rw interface* [interface]
|   |   |   |   +--rw interface          mpls-interface-ref
|   |   |   |   +--rw config
|   |   |   |   |   +--rw hello-holdtime?          uint16
|   |   |   |   |   +--rw hello-interval?          uint16
|   |   |   |   |   +--rw igp-synchronization-delay? uint16 {per-interface-ti
mer-config}?
|   |   |   +--rw address-family* [afi]
|   |   |   |   +--rw afi          ldp-address-family
|   |   |   |   +--rw config
|   |   |   |   |   +--rw enable?      boolean
|   |   |   |   |   +--rw ipv4
|   |   |   |   |   |   +--rw transport-address?      union
|   |   |   |   |   +--rw ipv6
|   |   |   |   |   |   +--rw transport-address?      union
|   |   +--rw targeted

```



```

+--rw config
|   +--rw hello-holdtime?    uint16
|   +--rw hello-interval?    uint16
|   +--rw hello-accept {policy-extended-discovery-config}?
|       +--rw enable?        boolean
|       +--rw neighbor-list?  neighbor-list-ref
+--rw address-family* [afi]
|   +--rw afi                ldp-address-family
|   +--rw ipv4
|       +--rw target* [adjacent-address]
|           +--rw adjacent-address    inet:ipv4-address
|           +--rw config
|               +--rw enable?        boolean
|               +--rw local-address?  inet:ipv4-address
|   +--rw ipv6
|       +--rw target* [adjacent-address]
|           +--rw adjacent-address    inet:ipv6-address
|           +--rw config
|               +--rw enable?        boolean
|               +--rw local-address?  inet:ipv6-address
+--rw forwarding-nexthop {forwarding-nexthop-config}?
|   +--rw interfaces
|       +--rw interface* [interface]
|           +--rw interface          mpls-interface-ref
|           +--rw address-family* [afi]
|               +--rw afi            ldp-address-family
|               +--rw config
|                   +--rw ldp-disable?  boolean
+--rw label-policy
|   +--rw independent-mode
|       +--rw assign {policy-label-assignment-config}?
|           +--rw (prefix-option)?
|               +--rw prefix-list?    prefix-list-ref
|               +--rw host-routes-only?  boolean
|       +--rw advertise
|           +--rw explicit-null
|               +--rw enable?        boolean
|               +--rw prefix-list?    prefix-list-ref
|           +--rw prefix-list?        prefix-list-ref
|       +--rw accept
|           +--rw prefix-list?        prefix-list-ref
+--rw ordered-mode {policy-ordered-label-config}?
|   +--rw egress-lsr
|       +--rw prefix-list?    prefix-list-ref
|   +--rw advertise
|       +--rw prefix-list?    prefix-list-ref
|   +--rw accept
|       +--rw prefix-list?    prefix-list-ref

```

```

+--rw peers
  +--rw config
    |   +--rw session-authentication-md5-password?  string
    |   +--rw session-ka-holdtime?                  uint16
    |   +--rw session-ka-interval?                  uint16
    |   +--rw session-downstream-on-demand {session-downstream-on-demand-con
fig}?
    |   |   +--rw enable?          boolean
    |   |   +--rw peer-list?      peer-list-ref
    +--rw peer* [lsr-id]
      +--rw lsr-id      yang:dotted-quad
      +--rw config
        +--rw admin-down?          boolean
        +--rw capability
        +--rw label-policy
          |   +--rw advertise
          |   |   +--rw prefix-list?  prefix-list-ref
          |   +--rw accept
          |   |   +--rw prefix-list?  prefix-list-ref
        +--rw session-authentication-md5-password?  string
        +--rw graceful-restart
          |   +--rw enable?          boolean
          |   +--rw reconnect-time?  uint16
          |   +--rw recovery-time?   uint16
        +--rw session-ka-holdtime?          uint16
        +--rw session-ka-interval?          uint16
        +--rw address-family
          +--rw ipv4
            |   +--rw label-policy
            |   |   +--rw advertise
            |   |   |   +--rw prefix-list?  prefix-list-ref
            |   |   +--rw accept
            |   |   |   +--rw prefix-list?  prefix-list-ref
          +--rw ipv6
            +--rw label-policy
            |   +--rw advertise
            |   |   +--rw prefix-list?  prefix-list-ref
            +--rw accept
            |   +--rw prefix-list?  prefix-list-ref

```

Figure 3

### 3.2.1. Configuration Hierarchy

The LDP configuration container is logically divided into following high-level config areas:

```
Per-VRF parameters
  o Global parameters
  o Per-address-family parameters
  o LDP Capabilities parameters
  o Hello Discovery parameters
    - interfaces
      - Per-interface:
        Global
        Per-address-family
    - targeted
      - Per-target
  o Peer parameters
    - Global
    - Per-peer
      Per-address-family
      Capabilities parameters
  o Forwarding parameters
```

Figure 4

Following subsections briefly explain these configuration areas.

#### 3.2.1.1. Per-VRF parameters

LDP module resides under an network-instance and the scope of any LDP configuration defined under this tree is per network-instance (per-VRF). This configuration is further divided into sub categories as follows.

##### 3.2.1.1.1. Per-VRF global parameters

There are configuration items that are available directly under a VRF instance and do not fall under any other sub tree. Example of such a parameter is LDP LSR id that is typically configured per VRF. To keep legacy LDP features and applications working in an LDP IPv4 networks with this model, this document recommends an operator to pick a routable IPv4 unicast address as an LSR Id.

##### 3.2.1.1.2. Per-VRF Capabilities parameters

This container falls under global tree and holds the LDP capabilities that are to be enabled for certain features. By default, an LDP capability is disabled unless explicitly enabled. These capabilities are typically used to negotiate with LDP peer(s) the support/non-support related to a feature and its parameters. The scope of a capability enabled under this container applies to all LDP peers in the given VRF instance. There is also a peer level capability

container that is provided to override a capability that is enabled/specified at VRF level.

#### 3.2.1.1.3. Per-VRF Per-Address-Family parameters

Any LDP configuration parameter related to IP address family (AF) whose scope is VRF wide is configured under this tree. The examples of per-AF parameters include enabling LDP for an address family, prefix-list based label policies, and LDP transport address.

#### 3.2.1.1.4. Per-VRF Hello Discovery parameters

This container is used to hold LDP configuration related to Hello and discovery process for both basic (link) and extended (targeted) discovery.

The "interfaces" is a container to configure parameters related to VRF interfaces. There are parameters that apply to all interfaces (such as hello timers), as well as parameters that can be configured per-interface. Hence, an interface list is defined under "interfaces" container. The model defines parameters to configure per-interface non AF related items, as well as per-interface per-AF items. The example of former is interface hello timers, and example of latter is enabling hellos for a given AF under an interface.

The "targeted" container under a VRF instance allows to configure LDP targeted discovery related parameters. Within this container, the "target" list provides a mean to configure multiple target addresses to perform extended discovery to a specific destination target, as well as to fine-tune the per-target parameters.

#### 3.2.1.1.5. Per-VRF Peer parameters

This container is used to hold LDP configuration related to LDP sessions and peers under a VRF instance. This container allows to configure parameters that either apply on VRF's all peers or a subset (peer-list) of VRF peers. The example of such parameters include authentication password, session KA timers etc. Moreover, the model also allows per-peer parameter tuning by specifying a "peer" list under the "peers" container. A peer is uniquely identified using its LSR Id and hence LSR Id is the key for peer list

Like per-interface parameters, some per-peer parameters are AF-agnostic (i.e. either non AF related or apply to both IP address families), and some that belong to an AF. The example of former is per-peer session password configuration, whereas the example of latter is prefix-list based label policies (inbound and outbound) that apply to a given peer.

### 3.2.1.1.6. Per-VRF Forwarding parameters

This container is used to hold configuration used to control LDP forwarding behavior under a VRF instance. One example of a configuration under this container is when a user wishes to enable neighbor discovery on an interface but wishes to disable use of the same interface as forwarding nexthop. This example configuration makes sense only when there are more than one LDP enabled interfaces towards the neighbor.

### 3.2.2. All-VRFs Configuration

[Ed note: TODO]

### 3.3. Operational State

Operational state of LDP can be queried and obtained from read-only state containers that fall under the same tree (/rt:routing/rt:control-plane-protocols/) as the configuration.

Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the protocol. [Ed note: This is where this model differs presently from [I-D.openconfig-netmod-opstate] and subject to alignment in later revisions]

Following is a simplified graphical representation of the data model for LDP operational state.

```

module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
  +--rw mpls-ldp!
    +--rw global
      +--ro state
        +--ro capability
          +--ro end-of-lib {capability-end-of-lib}?
            | +--ro enable?    boolean
          +--ro typed-wildcard-fec {capability-typed-wildcard-fec}?
            | +--ro enable?    boolean
          +--ro upstream-label-assignment {capability-upstream-label-assignment}?
            | +--ro enable?    boolean
          +--ro graceful-restart
            +--ro enable?      boolean
            +--ro helper-enable? boolean {graceful-restart-helper-mod
e}?
          +--ro reconnect-time?      uint16
          +--ro recovery-time?        uint16
          +--ro forwarding-holdtime?  uint16

```

```

+--ro igmp-synchronization-delay?    uint16
+--ro lsr-id?                        yang:dotted-quad
+--rw address-family* [afi]
+--rw afi                            ldp-address-family
+--ro state
+--ro enable?                        boolean
+--ro label-policy
+--ro independent-mode
+--ro assign {policy-label-assignment-config}?
+--ro (prefix-option)?
+--:(prefix-list)
+--ro prefix-list?                    prefix-list-ref
+--:(host-routes-only)
+--ro host-routes-only?              boolean
+--ro advertise
+--ro explicit-null
+--ro enable?                        boolean
+--ro prefix-list?                    prefix-list-ref
+--ro prefix-list?                    prefix-list-ref
+--ro accept
+--ro prefix-list?                    prefix-list-ref
+--ro ordered-mode {policy-ordered-label-config}?
+--ro egress-lsr
+--ro prefix-list?                    prefix-list-ref
+--ro advertise
+--ro prefix-list?                    prefix-list-ref
+--ro accept
+--ro prefix-list?                    prefix-list-ref
+--ro ipv4
+--ro transport-address?              inet:ipv4-address
+--ro bindings
+--ro address* [address]
+--ro address                        inet:ipv4-address
+--ro advertisement-type?             advertised-received
+--ro peer?                           leafref
+--ro fec-label* [fec]
+--ro fec                            inet:ipv4-prefix
+--ro peer* [peer advertisement-type]
+--ro peer                           leafref
+--ro advertisement-type              advertised-received
+--ro label?                          mpls:mpls-label
+--ro used-in-forwarding?              boolean
+--ro ipv6
+--ro transport-address?              inet:ipv6-address
+--ro binding
+--ro address* [address]
+--ro address                        inet:ipv6-address
+--ro advertisement-type?             advertised-received

```

```

|   |--ro peer?          leafref
|--ro fec-label* [fec]
|   |--ro fec            inet:ipv6-prefix
|--ro peer* [peer advertisement-type]
|   |--ro peer          leafref
|   |--ro advertisement-type advertised-received
|   |--ro label?        mpls:mpls-label
|   |--ro used-in-forwarding? boolean
|--rw discovery
|   |--rw interfaces
|       |--ro state
|           |--ro hello-holdtime?  uint16
|           |--ro hello-interval?  uint16
|--rw interface* [interface]
|   |--ro state
|       |--ro hello-holdtime?      uint16
|       |--ro hello-interval?      uint16
|       |--ro igp-synchronization-delay? uint16 {per-interface-ti
mer-config}?
|           |--ro next-hello?      uint16
|--rw address-family* [afi]
|   |--rw afi          ldp-address-family
|--ro state
|   |--ro enable?      boolean
|--ro ipv4
|   |--ro transport-address? union
|   |--ro hello-adjacencies* [adjacent-address]
|       |--ro adjacent-address    inet:ipv4-address
|       |--ro flag*              identityref
|       |--ro hello-holdtime
|           |--ro adjacent?      uint16
|           |--ro negotiated?    uint16
|           |--ro remaining?     uint16
|       |--ro next-hello?        uint16
|--ro statistics
|   |--ro discontinuity-time      yang:date-and-time
|   |--ro hello-received?        yang:counter64
|   |--ro hello-dropped?         yang:counter64
|--ro peer?          leafref
|--ro ipv6
|   |--ro transport-address? union
|   |--ro hello-adjacencies* [adjacent-address]
|       |--ro adjacent-address    inet:ipv6-address
|       |--ro flag*              identityref
|       |--ro hello-holdtime
|           |--ro adjacent?      uint16
|           |--ro negotiated?    uint16
|           |--ro remaining?     uint16
|       |--ro next-hello?        uint16

```

```

| | | +---ro statistics
| | | |   |--ro discontinuity-time      yang:date-and-time
| | | |   |--ro hello-received?        yang:counter64
| | | |   |--ro hello-dropped?         yang:counter64
| | | +---ro peer?                     leafref
+--rw targeted
| +--ro state
| |   |--ro hello-holdtime?            uint16
| |   |--ro hello-interval?           uint16
| |   +---ro hello-accept {policy-extended-discovery-config}?
| |     |--ro enable?                  boolean
| |     |--ro neighbor-list?          neighbor-list-ref
+--rw address-family* [afi]
| +--rw afi                ldp-address-family
| +--ro state
| |   |--ro ipv4
| | |   +---ro hello-adjacencies* [local-address adjacent-address
| | | |   |--ro local-address              inet:ipv4-address
| | | |   |--ro adjacent-address             inet:ipv4-address
| | | |   |--ro flag*                       identityref
| | | |   +---ro hello-holdtime
| | | | |   |--ro adjacent?                 uint16
| | | | |   |--ro negotiated?               uint16
| | | | |   |--ro remaining?               uint16
| | | | +---ro next-hello?                   uint16
| | | | +---ro statistics
| | | | |   |--ro discontinuity-time        yang:date-and-time
| | | | |   |--ro hello-received?          yang:counter64
| | | | |   |--ro hello-dropped?          yang:counter64
| | | | +---ro peer?                         leafref
+--ro ipv6
| +---ro hello-adjacencies* [local-address adjacent-address
| |   |--ro local-address              inet:ipv6-address
| |   |--ro adjacent-address             inet:ipv6-address
| |   |--ro flag*                       identityref
| |   +---ro hello-holdtime
| | |   |--ro adjacent?                 uint16
| | |   |--ro negotiated?               uint16
| | |   |--ro remaining?               uint16
| | | +---ro next-hello?                   uint16
| | | +---ro statistics
| | | |   |--ro discontinuity-time        yang:date-and-time
| | | |   |--ro hello-received?          yang:counter64
| | | |   |--ro hello-dropped?          yang:counter64
| | | +---ro peer?                         leafref
+--rw ipv4
| +--rw target* [adjacent-address]
| |   +---ro adjacent-address             inet:ipv4-address

```



```

|         |         |         +---ro state                boolean
|         |         |         +---ro enable?              boolean
|         |         |         +---ro local-address?       inet:ipv4-address
+---rw ipv6
|         |         |         +---rw target* [adjacent-address]
|         |         |         +---rw adjacent-address      inet:ipv6-address
|         |         |         +---ro state
|         |         |         +---ro enable?              boolean
|         |         |         +---ro local-address?       inet:ipv6-address
+---rw forwarding-nexthop {forwarding-nexthop-config}?
|         |         +---rw interfaces
|         |         |         +---rw interface* [interface]
|         |         |         +---rw interface            mpls-interface-ref
|         |         |         +---rw address-family* [afi]
|         |         |         +---rw afi                  ldp-address-family
|         |         |         +---ro state
|         |         |         +---ro ldp-disable?        boolean
+---rw peers
|         |         +---ro state
|         |         |         +---ro session-authentication-md5-password? string
|         |         |         +---ro session-ka-holdtime?   uint16
|         |         |         +---ro session-ka-interval?   uint16
|         |         |         +---ro session-downstream-on-demand {session-downstream-on-demand-con
fig)?
|         |         |         +---ro enable?              boolean
|         |         |         +---ro peer-list?          peer-list-ref
+---rw peer* [lsr-id]
|         |         |         +---rw lsr-id             yang:dotted-quad
|         |         |         +---ro state
|         |         |         +---ro admin-down?          boolean
|         |         |         +---ro capability
|         |         |         +---ro label-policy
|         |         |         |         +---ro advertise
|         |         |         |         |         +---ro prefix-list?    prefix-list-ref
|         |         |         |         +---ro accept
|         |         |         |         +---ro prefix-list?    prefix-list-ref
+---ro session-authentication-md5-password? string
+---ro graceful-restart
|         |         |         +---ro enable?              boolean
|         |         |         +---ro reconnect-time?      uint16
|         |         |         +---ro recovery-time?       uint16
+---ro session-ka-holdtime?           uint16
+---ro session-ka-interval?          uint16
+---ro address-family
|         |         |         +---ro ipv4
|         |         |         |         +---ro label-policy
|         |         |         |         |         +---ro advertise
|         |         |         |         |         |         +---ro prefix-list?    prefix-list-ref
|         |         |         |         +---ro accept

```

```

|      +--ro prefix-list?    prefix-list-ref
+--ro hello-adjacencies* [local-address adjacent-address]
|   +--ro local-address      inet:ipv4-address
|   +--ro adjacent-address   inet:ipv4-address
|   +--ro flag*              identityref
|   +--ro hello-holdtime
|   |   +--ro adjacent?      uint16
|   |   +--ro negotiated?    uint16
|   |   +--ro remaining?     uint16
|   +--ro next-hello?        uint16
|   +--ro statistics
|   |   +--ro discontinuity-time    yang:date-and-time
|   |   +--ro hello-received?      yang:counter64
|   |   +--ro hello-dropped?       yang:counter64
|   +--ro interface?            mpls-interface-ref
+--ro ipv6
|   +--ro label-policy
|   |   +--ro advertise
|   |   |   +--ro prefix-list?    prefix-list-ref
|   |   +--ro accept
|   |   |   +--ro prefix-list?    prefix-list-ref
|   +--ro hello-adjacencies* [local-address adjacent-address]
|   |   +--ro local-address      inet:ipv6-address
|   |   +--ro adjacent-address   inet:ipv6-address
|   |   +--ro flag*              identityref
|   |   +--ro hello-holdtime
|   |   |   +--ro adjacent?      uint16
|   |   |   +--ro negotiated?    uint16
|   |   |   +--ro remaining?     uint16
|   |   +--ro next-hello?        uint16
|   |   +--ro statistics
|   |   |   +--ro discontinuity-time    yang:date-and-time
|   |   |   +--ro hello-received?      yang:counter64
|   |   |   +--ro hello-dropped?       yang:counter64
|   |   +--ro interface?            mpls-interface-ref
+--ro label-advertisement-mode
|   +--ro local?              label-adv-mode
|   +--ro peer?               label-adv-mode
|   +--ro negotiated?         label-adv-mode
+--ro next-keep-alive?        uint16
+--ro peer-ldp-id?            yang:dotted-quad
+--ro received-peer-state
|   +--ro graceful-restart
|   |   +--ro enable?          boolean
|   |   +--ro reconnect-time?  uint16
|   |   +--ro recovery-time?   uint16
+--ro capability
+--ro end-of-lib

```

```

|      |  +--ro enable?    boolean
|      +--ro typed-wildcard-fec
|      |  +--ro enable?    boolean
|      +--ro upstream-label-assignment
|      +--ro enable?    boolean
+--ro session-holdtime
|  +--ro peer?          uint16
|  +--ro negotiated?    uint16
|  +--ro remaining?     uint16
+--ro session-state?          enumeration
+--ro tcp-connection
|  +--ro local-address?      inet:ip-address
|  +--ro local-port?         inet:port-number
|  +--ro remote-address?     inet:ip-address
|  +--ro remote-port?        inet:port-number
+--ro up-time?              string
+--ro statistics
|  +--ro discontinuity-time      yang:date-and-time
|  +--ro received
|  |  +--ro total-octets?      yang:counter64
|  |  +--ro total-messages?   yang:counter64
|  |  +--ro address?          yang:counter64
|  |  +--ro address-withdraw? yang:counter64
|  |  +--ro initialization?    yang:counter64
|  |  +--ro keepalive?        yang:counter64
|  |  +--ro label-abort-request? yang:counter64
|  |  +--ro label-mapping?    yang:counter64
|  |  +--ro label-release?     yang:counter64
|  |  +--ro label-request?     yang:counter64
|  |  +--ro label-withdraw?    yang:counter64
|  |  +--ro notification?     yang:counter64
|  +--ro sent
|  |  +--ro total-octets?      yang:counter64
|  |  +--ro total-messages?   yang:counter64
|  |  +--ro address?          yang:counter64
|  |  +--ro address-withdraw? yang:counter64
|  |  +--ro initialization?    yang:counter64
|  |  +--ro keepalive?        yang:counter64
|  |  +--ro label-abort-request? yang:counter64
|  |  +--ro label-mapping?    yang:counter64
|  |  +--ro label-release?     yang:counter64
|  |  +--ro label-request?     yang:counter64
|  |  +--ro label-withdraw?    yang:counter64
|  |  +--ro notification?     yang:counter64
+--ro total-addresses?      uint32
+--ro total-labels?         uint32
+--ro total-fec-label-bindings? uint32

```

Figure 5

### 3.3.1. Derived States

Following are main areas for which LDP operational "derived" state is defined:

- Neighbor Adjacencies

- Peer

- Bindings (FEC-label and address)

- Capabilities

#### 3.3.1.1. Adjacency state

Neighbor adjacencies are per address-family hello adjacencies that are formed with neighbors as result of LDP basic or extended discovery. In terms of organization, there is a source of discovery (e.g. interface or target address) along with its associated parameters and one or more discovered neighbors along with neighbor discovery related parameters. For the basic discovery, there could be more than one discovered neighbor for a given source (interface), whereas there is at most one discovered neighbor for an extended discovery source (local-address and target-address). This is also to be noted that the reason for a targeted neighbor adjacency could be either an active source (locally configured targeted) or passive source (to allow any incoming extended/targeted hellos). A neighbor/adjacency record also contains session-state that helps highlight whether a given adjacency has progressed to subsequent session level or to eventual peer level.

Following captures high level tree hierarchy for neighbor adjacency state.

```

+--rw mpls-ldp!
+--rw discovery
+--rw interfaces
|   +--rw interface* [interface]
|       +--rw address-family* [af]
|           +--ro state
|               +--ro ipv4 (or ipv6)
|                   +--ro hello-adjacencies* [adjacent-address]
|                       +--ro adjacent-address
|                           . . . .
|                           . . . .
+--rw targeted
+--rw address-family* [afi]
+--rw afi          address-family
+--ro state
+--ro ipv4 (or ipv6)
+--ro hello-adjacencies* [local-address adjacent-addresses]
    +--ro local-address
    +--ro adjacent-address
        . . . .
        . . . .

```

Figure 6

### 3.3.1.2. Peer state

Peer related derived state is presented under peers tree. This is one of the core state that provides info on the session related parameters (mode, authentication, KA timeout etc.), TCP connection info, hello adjacencies for the peer, statistics related to messages and bindings, and capabilities exchange info.

Following captures high level tree hierarchy for peer state.

```

+--rw mpls-ldp!
  +--rw peers
    +--rw peer* [lsr-id]
      +--rw lsr-id
      +--ro state
        +--ro session-ka-holdtime?
        +-- . . . .
        +-- . . . .
        +--ro capability
        + +ro -- . . .
        +--ro address-family
          +--ro ipv4 (or ipv6)
            +--ro hello-adjacencies* [local-address adjacent-address]
            . . . .
            . . . .
        +--ro received-peer-state
          +--ro . . . .
          +--ro capability
          +--ro . . . .
        +--ro statistics
          +-- . . . .
          +-- . . . .

```

Figure 7

### 3.3.1.3. Bindings state

Binding state provides information on LDP FEC-label bindings as well as address binding for both inbound (received) as well as outbound (advertised) direction. FEC-label bindings are presented as a FEC-centric view, and address bindings are presented as an address-centric view:

```

FEC-Label bindings:
  FEC 200.1.1.1/32:
    advertised: local-label 16000
    peer 192.168.0.2:0
    peer 192.168.0.3:0
    peer 192.168.0.4:0
    received:
      peer 192.168.0.2:0, label 16002, used-in-forwarding=Yes
      peer 192.168.0.3:0, label 17002, used-in-forwarding=No
  FEC 200.1.1.2/32:
    . . . .
  FEC 201.1.0.0/16:
    . . . .

Address bindings:
  Addr 1.1.1.1:
    advertised
  Addr 1.1.1.2:
    advertised
  Addr 2.2.2.2:
    received, peer 192.168.0.2
  Addr 2.2.2.22:
    received, peer 192.168.0.2
  Addr 3.3.3.3:
    received, peer 192.168.0.3
  Addr 3.3.3.33:
    received, peer 192.168.0.3

```

Figure 8

Note that all local addresses are advertised to all peers and hence no need to provide per-peer information for local address advertisement. Furthermore, note that it is easy to derive a peer-centric view for the bindings from the information already provided in this model.

Following captures high level tree hierarchy for bindings state.

```

+--rw mpls-ldp!
  +--rw global
    +--rw address-family* [afi]
      +--rw afi          address-family
      +--ro state
        +--ro ipv4 (or ipv6)
          +--ro bindings
            +--ro address* [address]
              | +--ro address
              | +--ro direction?   advertised-received
              | +--ro peer?        leafref
            +--ro fec-label* [fec]
              +--ro fec          inet:ipv4-prefix
              +--ro peer* [peer advertisement-type]
                +--ro peer          leafref
                +--ro advertisement-type   advertised-received
                +--ro label?          mpls:mpls-label
                +--ro used-in-forwarding?  boolean

```

Figure 9

#### 3.3.1.4. Capabilities state

LDP capabilities state comprise two types of information - global information (such as timer etc.), and per-peer information.

Following captures high level tree hierarchy for LDP capabilities state.

```

+--rw mpls-ldp!
  +--rw global
    | +--ro state
    |   +--ro capability
    |     +--ro . . . .
    |     +--ro . . . .
  +--rw peers
    +--rw peer* [lsr-id]
      +--rw lsr-id    yang:dotted-quad
      +--ro state
        +--ro received-peer-state
        +--ro capability
          +--ro . . . .
          +--ro . . . .

```

Figure 10



### 3.4. Notifications

This model defines a list of notifications to inform client of important events detected during the protocol operation. These events include events related to changes in the operational state of an LDP peer, hello adjacency, and FEC etc. It is to be noted that an LDP FEC is treated as operational (up) as long as it has at least 1 NHLFE with outgoing label.

Following is a simplified graphical representation of the data model for LDP notifications.

```

module: ietf-mpls-ldp
notifications:
  +---n mpls-ldp-peer-event
  |   +--ro event-type?   oper-status-event-type
  |   +--ro peer-ref?    leafref
  +---n mpls-ldp-hello-adjacency-event
  |   +--ro event-type?   oper-status-event-type
  |   +--ro (hello-adjacency-type)?
  |   |   +--:(targeted)
  |   |   |   +--ro targeted
  |   |   |   +--ro target-address?   inet:ip-address
  |   |   +--:(link)
  |   |   |   +--ro link
  |   |   |   +--ro next-hop-interface?   mpls-interface-ref
  |   |   |   +--ro next-hop-address?    inet:ip-address
  +---n mpls-ldp-fec-event
  |   +--ro event-type?   oper-status-event-type
  |   +--ro prefix?       inet:ip-prefix

```

Figure 11

### 3.5. Actions

This model defines a list of rpcs that allow performing an action or executing a command on the protocol. For example, it allows to clear (reset) LDP peers, hello-adjacencies, and statistics. The model makes an effort to provide different level of control so that a user is able to either clear all, or clear all for a given type, or clear a specific entity.

Following is a simplified graphical representation of the data model for LDP actions.

```

module: ietf-mpls-ldp
rpcs:
  +---x mpls-ldp-clear-peer
  |   +---w input
  |   +---w lsr-id?    union
  +---x mpls-ldp-clear-hello-adjacency
  |   +---w input
  |   +---w hello-adjacency
  |   +---w (hello-adjacency-type)?
  |   |   +---:(targeted)
  |   |   |   +---w targeted!
  |   |   |   +---w target-address?    inet:ip-address
  |   |   +---:(link)
  |   |   |   +---w link!
  |   |   |   +---w next-hop-interface?    mpls-interface-ref
  |   |   |   +---w next-hop-address?     inet:ip-address
  +---x mpls-ldp-clear-peer-statistics
  |   +---w input
  |   +---w lsr-id?    union

```

Figure 12

#### 4. mLDP YANG Model

##### 4.1. Overview

Due to tight dependency of mLDP on LDP, mLDP model builds on top of LDP model defined earlier in the document. Following are the main mLDP areas and documents that are within the scope of this model:

- o mLDP Base Specification [RFC6388]
- o mLDP Recursive FEC [RFC6512]
- o Targeted mLDP [RFC7060]
- o mLDP Fast-Reroute (FRR)
  - \* Node Protection [RFC7715]
  - \* Multicast-only
- o Hub-and-Spoke Multipoint LSPs [RFC7140]
- o mLDP In-band Signaling [RFC6826] (future revision)
- o mLDP In-band signaling in a VRF [RFC7246]

- o mLDP In-band Signaling with Wildcards [RFC7438] (future revision)
- o Configured Leaf LSPs (manually provisioned)

[Ed Note: Some of the topics in the above list are to be addressed/added in later revision of this document].

## 4.2. Configuration

### 4.2.1. Configuration Hierarchy

In terms of overall configuration layout, following figure highlights extensions to LDP configuration model to incorporate mLDP:

```

+-- mpls-ldp
+-- ...
+-- ...
+-- mldp
+-- ...
+-- ...
+-- address-family* [af]
+-- af
+-- ...
+-- ...
+-- global
+-- ...
+-- capability
+-- ...
+-- ...
+-- mldp
+-- ...
+-- ...
+-- discovery
+-- ...
+-- ...
+-- forwarding-nexthop
+-- interfaces
+-- interface* [interface]
+-- interface
+-- address-family* [af]
+-- af
+-- ...
+-- mldp-disable
+-- peers
+-- ...
+-- ...
+-- peer* [lsr-id]
+-- ...
+-- ...
+-- capability
+-- ...
+-- ...
+-- mldp
+-- ...
+-- ...

```

Figure 13

From above hierarchy, we can categorize mLDP configuration parameters into two types:

- o Parameters that leverage/extend LDP containers and parameters
- o Parameters that are mLDP specific

Following subsections first describe mLDP specific configuration parameters, followed by those leveraging LDP.

#### 4.2.2. mldp container

mldp container resides directly under "mpls-ldp" and holds the configuration related to items that are mLDP specific. The main items under this container are:

- o mLDP enabling: To enable mLDP under a (VRF) routing instance, mldp container is enabled under LDP. Given that mLDP requires LDP signalling, it is not sensible to allow disabling LDP control plane under a (VRF) network-instance while requiring mLDP to be enabled for the same. However, if a user wishes only to allow signalling for multipoint FECs on an LDP/mLDP enabled VRF instance, he/she can use LDP label-policies to disable unicast FECs under the VRF.
- o mLDP per-AF features: mLDP manages its own list of IP address-families and the features enabled underneath. The per-AF mLDP configuration items include:
  - \* Multicast-only FRR: This enables Multicast-only FRR functionality for a given AF under mLDP. The feature allows route-policy to be configured for finer control/applicability of the feature.
  - \* Recursive FEC: The recursive-fec feature [RFC6512] can be enabled per AF with a route-policy.
  - \* Configured Leaf LSPs: To provision multipoint leaf LSP manually, a container is provided per-AF under LDP. The configuration is flexible and allows a user to specify MP LSPs of type p2mp or mp2mp with IPv4 or IPv6 root address(es) by using either LSP-Id or (S,G).

Targeted mLDP feature specification [RFC7060] do not require any mLDP specific configuration. It, however, requires LDP upstream-label-assignment capability [RFC6389] to be enabled.

#### 4.2.3. Leveraging LDP containers

mLDP configuration model leverages following configuration areas and containers that are already defined for LDP:

- o Capabilities: A new container "mldp" is defined under Capabilities container. This new container specifies any mLDP specific capabilities and their parameters. Moreover, a new "mldp" container is also added under per-peer capability container to override/control mLDP specific capabilities on a peer level. In the scope of this document, the most important capabilities related to mLDP are p2mp, mp2mp, make-before-break, hub-and-spoke, and node-protection.
- o Discovery and Peer: mLDP requires LDP discovery and peer procedures to form mLDP peering. A peer is treated as mLDP peer only when either P2MP or MP2MP capabilities have been successfully exchanged with the peer. If a user wish to selectively enable or disable mLDP with a LDP-enabled peer, he/she may use per-peer mLDP capabilities configuration. [Ed Note: The option to control mLDP enabling/disabling on a peer-list is being explored for future ]. In most common deployments, it is desirable to disable mLDP (capabilities announcements) on a targeted-only LDP peering, where targeted-only peer is the one whose discovery sources are targeted only. In future revision, a configuration option for this support will also be provided.
- o Forwarding: By default, mLDP is allowed to select any of the LDP enabled interface as a downstream interface towards a nexthop (LDP/mLDP peer) for MP LSP programming. However, a configuration option is provided to allow mLDP to exclude a given interface from such a selection. Note that such a configuration option will be useful only when there are more than one interfaces available for the downstream selection.

This goes without saying that mLDP configuration tree follows the same approach as LDP, where the tree comprise leafs for intended configuration.

#### 4.2.4. YANG tree

The following figure captures the YANG tree for mLDP configuration. To keep the focus, the figure has been simplified to display only mLDP items without any LDP items.

```
module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
  +--rw mpls-ldp!
```

```

+--rw global
|
|   +--rw config
|   |
|   |   +--rw capability
|   |   |
|   |   |   +--rw mldp {mldp}?
|   |   |   |
|   |   |   |   +--rw p2mp
|   |   |   |   |
|   |   |   |   |   +--rw enable?    boolean
|   |   |   |   |
|   |   |   |   +--rw mp2mp
|   |   |   |   |
|   |   |   |   |   +--rw enable?    boolean
|   |   |   |   |
|   |   |   |   +--rw make-before-break
|   |   |   |   |
|   |   |   |   |   +--rw enable?    boolean
|   |   |   |   |   +--rw switchover-delay? uint16
|   |   |   |   |   +--rw timeout?    uint16
|   |   |   |   +--rw hub-and-spoke {capability-mldp-hsmp}?
|   |   |   |   |
|   |   |   |   |   +--rw enable?    boolean
|   |   |   |   +--rw node-protection {capability-mldp-node-protection}?
|   |   |   |   |
|   |   |   |   |   +--rw plr?        boolean
|   |   |   |   |   +--rw merge-point
|   |   |   |   |   |
|   |   |   |   |   |   +--rw enable?    boolean
|   |   |   |   |   |   +--rw targeted-session-teardown-delay? uint16
|   |   |   +--rw mldp {mldp}?
|   |   |   |
|   |   |   |   +--rw config
|   |   |   |   |
|   |   |   |   |   +--rw enable?    boolean
|   |   |   |   +--rw address-family* [afi]
|   |   |   |   |
|   |   |   |   |   +--rw afi
|   |   |   |   |   |
|   |   |   |   |   |   ldp-address-family
|   |   |   |   |   +--rw config
|   |   |   |   |   |
|   |   |   |   |   |   +--rw multicast-only-frr {mldp-mofrr}?
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--rw prefix-list?    prefix-list-ref
|   |   |   |   |   |   |
|   |   |   |   |   |   +--rw recursive-fec
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--rw prefix-list?    prefix-list-ref
|   |   |   |   +--rw configured-leaf-lsps
|   |   |   |   |
|   |   |   |   |   +--rw p2mp
|   |   |   |   |   |
|   |   |   |   |   |   +--rw roots-ipv4
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--rw root* [root-address]
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   +--rw root-address    inet:ipv4-address
|   |   |   |   |   |   |   |   +--rw lsp* [lsp-id source-address group-address]
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   +--rw lsp-id        uint16
|   |   |   |   |   |   |   |   |   +--rw source-address  inet:ipv4-address
|   |   |   |   |   |   |   |   |   +--rw group-address   inet:ipv4-address-no-zone
|   |   |   |   |   |   +--rw roots-ipv6
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--rw root* [root-address]
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   +--rw root-address    inet:ipv6-address
|   |   |   |   |   |   |   |   +--rw lsp* [lsp-id source-address group-address]
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   +--rw lsp-id        uint16
|   |   |   |   |   |   |   |   |   +--rw source-address  inet:ipv6-address
|   |   |   |   |   |   |   |   |   +--rw group-address   inet:ipv6-address-no-zone
|   |   |   |   +--rw mp2mp
|   |   |   |   |
|   |   |   |   |   +--rw roots-ipv4
|   |   |   |   |   |
|   |   |   |   |   |   +--rw root* [root-address]

```

```

|--rw root-address          inet:ipv4-address
+--rw lsp* [lsp-id source-address group-address]
    |--rw lsp-id            uint16
    |--rw source-address     inet:ipv4-address
    |--rw group-address      inet:ipv4-address-no-zone
+--rw roots-ipv6
    |--rw root* [root-address]
    |--rw root-address       inet:ipv6-address
    |--rw lsp* [lsp-id source-address group-address]
        |--rw lsp-id        uint16
        |--rw source-address inet:ipv6-address
        |--rw group-address  inet:ipv6-address-no-zone
+--rw forwarding-nexthop {forwarding-nexthop-config}?
    +--rw interfaces
        +--rw interface* [interface]
            |--rw interface      if:interface-ref
            |--rw address-family* [afi]
                |--rw afi        address-family
                |--rw config
                    |--rw mldp-disable?    boolean {mldp}?
+--rw peers
    +--rw peer* [lsr-id]
        +--rw capability
            +--rw mldp {mldp}?
                +--rw p2mp
                    |--rw enable?    boolean
                +--rw mp2mp
                    |--rw enable?    boolean
            +--rw make-before-break
                |--rw enable?        boolean
                |--rw switchover-delay? uint16
                |--rw timeout?       uint16
            +--rw hub-and-spoke {capability-mldp-hsmp}?
                |--rw enable?        boolean
            +--rw node-protection {capability-mldp-node-protection}?
                +--rw plr?            boolean
            +--rw merge-point
                +--rw enable?        boolean
            +--rw targeted-session-teardown-delay? uint16

```

Figure 14

### 4.3. Operational State

Operational state of mLDP can be queried and obtained from this read-only container "mldp" which resides under mpls-ldp container.



Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the mLDP protocol.

Following is a simplified graphical representation of the data model for mLDP operational state:

```

module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
  +--rw mpls-ldp!
    +--rw global
      +--ro state
        +--ro capability
          +--ro mldp {mldp}?
            +--ro p2mp
              | +--ro enable?    boolean
            +--ro mp2mp
              | +--ro enable?    boolean
            +--ro make-before-break
              | +--ro enable?          boolean
              | +--ro switchover-delay? uint16
              | +--ro timeout?        uint16
            +--ro hub-and-spoke {capability-mldp-hsmp}?
              | +--ro enable?    boolean
            +--ro node-protection {capability-mldp-node-protection}?
              +--ro plr?          boolean
              +--ro merge-point
                +--ro enable?          boolean
                +--ro targeted-session-teardown-delay? uint16
          +--rw mldp {mldp}?
            +--ro state
              +--ro enable?    boolean
            +--rw address-family* [afi]
              +--rw afi
                ldp-address-family
              +--ro state
                +--ro multicast-only-frr {mldp-mofrr}?
                  | +--ro prefix-list?    prefix-list-ref
                +--ro recursive-fec
                  | +--ro prefix-list?    prefix-list-ref
                +--ro ipv4
                  +--ro roots
                    +--ro root* [root-address]
                      +--ro root-address    inet:ipv4-address
                      +--ro is-self?        boolean
                      +--ro reachability* [address interface]
                        +--ro address        inet:ipv4-address
                        +--ro interface      mpls-interface-ref

```

```

| | | | | +--ro peer? leafref
|--ro bindings
+--ro opaque-type-lspid
|   +--ro fec-label* [root-address lsp-id recur-root-addr
ess recur-rd]
|       +--ro root-address inet:ipv4-address
|       +--ro lsp-id uint32
|       +--ro recur-root-address inet:ip-address
|       +--ro recur-rd route-distinguisher
|       +--ro multipoint-type? multipoint-type
|       +--ro peer* [direction peer advertisement-type]
|           +--ro direction downstream-upstream
|           +--ro peer leafref
|           +--ro advertisement-type advertised-received
|           +--ro label? mpls:mpls-label
|           +--ro mbb-role? enumeration
|           +--ro mofrr-role? enumeration
+--ro opaque-type-src
+--ro fec-label* [root-address source-address group-a
ddress rd recur-root-address recur-rd]
|       +--ro root-address inet:ipv4-address
|       +--ro source-address inet:ip-address
|       +--ro group-address inet:ip-address-no-zon
e
|       +--ro rd route-distinguisher
|       +--ro recur-root-address inet:ip-address
|       +--ro recur-rd route-distinguisher
|       +--ro multipoint-type? multipoint-type
|       +--ro peer* [direction peer advertisement-type]
|           +--ro direction downstream-upstream
|           +--ro peer leafref
|           +--ro advertisement-type advertised-received
|           +--ro label? mpls:mpls-label
|           +--ro mbb-role? enumeration
|           +--ro mofrr-role? enumeration
+--ro opaque-type-bidir
+--ro fec-label* [root-address rp group-address rd re
cur-root-address recur-rd]
|       +--ro root-address inet:ipv4-address
|       +--ro rp inet:ip-address
|       +--ro group-address inet:ip-address-no-zon
e
|       +--ro rd route-distinguisher
|       +--ro recur-root-address inet:ip-address
|       +--ro recur-rd route-distinguisher
|       +--ro multipoint-type? multipoint-type
|       +--ro peer* [direction peer advertisement-type]
|           +--ro direction downstream-upstream
|           +--ro peer leafref
|           +--ro advertisement-type advertised-received
|           +--ro label? mpls:mpls-label
|           +--ro mbb-role? enumeration
|           +--ro mofrr-role? enumeration

```



```

+---ro multipoint-type?      multipoint-type
+---ro peer* [direction peer advertisement-type]
    +---ro direction          downstream-upstream
    +---ro peer                leafref
    +---ro advertisement-type  advertised-received
    +---ro label?              mpls:mpls-label
    +---ro mbb-role?           enumeration
    +---ro mofrr-role?         enumeration
+---rw forwarding-nexthop {forwarding-nexthop-config}?
    +---rw interfaces
        +---rw interface* [interface]
            +---rw address-family* [afi]
                +---ro state
                    +---ro mldp-disable?    boolean {mldp}?
+---rw peers
    +---rw peer* [lsr-id]
        +---ro state
            +---ro capability
                +---ro mldp {mldp}?
                +---ro p2mp
                    | +---ro enable?    boolean
                +---ro mp2mp
                    | +---ro enable?    boolean
                +---ro make-before-break
                    | +---ro enable?    boolean
                    | +---ro switchover-delay?  uint16
                    | +---ro timeout?      uint16
                +---ro hub-and-spoke {capability-mldp-hsmp}?
                    | +---ro enable?    boolean
                +---ro node-protection {capability-mldp-node-protection}?
                    +---ro plr?          boolean
                    +---ro merge-point
                        +---ro enable?    boolean
                    +---ro targeted-session-teardown-delay?  uint16
            +---ro received-peer-state
                +---ro capability
                    +---ro mldp {mldp}?
                    +---ro p2mp
                        | +---ro enable?    boolean
                    +---ro mp2mp
                        | +---ro enable?    boolean
                    +---ro make-before-break
                        | +---ro enable?    boolean
                    +---ro hub-and-spoke
                        | +---ro enable?    boolean
                    +---ro node-protection
                        +---ro plr?          boolean
                        +---ro merge-point?  boolean

```

Figure 15

#### 4.3.1. Derived states

Following are main areas for which mLDP operational derived state is defined:

- o Root
- o Bindings (FEC-label)
- o Capabilities

##### 4.3.1.1. Root state

Root address is a fundamental construct for MP FEC bindings and LSPs. The root state provides information on all the known roots in a given address-family, and their information on the root reachability (as learnt from RIB). In case of multi-path reachability to a root, the selection of upstream path is done on per-LSP basis at the time of LSP setup. Similarly, when protection mechanisms like MBB or MoFRR are in place, the path designation as active/standby or primary/backup is also done on per LSP basis. It is to be noted that a given root can be shared amongst multiple P2MP and/or MP2MP LSPs. Moreover, an LSP can be signaled to more than one root for RNR purposes.

The following diagram illustrates a root database on a branch/transit LSR:

```
root 1.1.1.1:
  path1:
    RIB: GigEthernet 1/0, 12.1.0.2;
    LDP: peer 192.168.0.1:0
  path2:
    RIB: GigEthernet 2/0, 12.2.0.2;
    LDP: peer 192.168.0.3:0

root 2.2.2.2:
  path1:
    RIB: 3.3.3.3;                (NOTE: This is a recursive path)
    LDP: peer 192.168.0.3:0      (NOTE: T-mLDP peer)

root 9.9.9.9:
  . . . .
```

Figure 16

A root entry on a root LSR itself will be presented as follows:

```
root 9.9.9.9:
  is-self
```

Figure 17

#### 4.3.1.2. Bindings state

Binding state provides information on mLDP FEC-label bindings for both P2MP and MP2MP FEC types. Like LDP, the FEC-label binding derived state is presented in a FEC-centric view per address-family, and provides information on both inbound (received) and outbound (advertised) bindings. The FEC is presented as (root-address, opaque-type-data) and the direction (upstream or downstream) is picked with respect to root reachability. In case of MBB or/and MoFRR, the role of a given peer binding is also provided with respect to MBB (active or standby) or/and MoFRR (primary or backup).

This document covers following type of opaque values with their keys in the operational model of mLDP bindings:

| Opaque Type            | Key               | RFC       |
|------------------------|-------------------|-----------|
| Generic LSP Identifier | LSP Id            | [RFC6388] |
| Transit IPv4 Source    | Source, Group     | [RFC6826] |
| Transit IPv6 Source    | Source, Group     | [RFC6826] |
| Transit IPv4 Bidir     | RP, Group         | [RFC6826] |
| Transit IPv6 Bidir     | RP, Group         | [RFC6826] |
| Transit VPNv4 Source   | Source, Group, RD | [RFC7246] |
| Transit VPNv6 Source   | Source, Group, RD | [RFC7246] |
| Transit VPNv4 Bidir    | RP, Group, RD     | [RFC7246] |
| Transit VPNv6 Bidir    | RP, Group, RD     | [RFC7246] |
| Recursive Opaque       | Root              | [RFC6512] |
| VPN-Recursive Opaque   | Root, RD          | [RFC6512] |

Table 1: MP Opaque Types and keys

It is to be noted that there are three basic types (LSP Id, Source, and Bidir) and then there are variants (VPN, recursive, VPN-recursive) on top of these basic types.

Following captures high level tree hierarchy for mLDP bindings state:

```

+--rw mpls-ldp!
  +--rw mldp
    +--rw address-family* [afi]
      +--rw afi          address-family
      +--ro state
        +--ro ipv4 (or ipv6)
          +--ro bindings
            +--ro opaque-type-xxx [root-address, type-specific-key]
              +--ro root-address
              +--ro ...
              +--ro recur-root-address      inet:ipv4-address
              +--ro recur-rd                route-distinguisher
              +--ro multipoint-type?        multipoint-type
              +--ro peer* [direction peer advertisement-type]
                +--ro direction            downstream-upstream
                +--ro peer                  leafref
                +--ro advertisement-type    advertised-received
                +--ro label?                mpls:mpls-label
                +--ro mbb-role?             enumeration
                +--ro mofrr-role?           enumeration

```

Figure 18

In the above tree, the type-specific-key varies with the base type as listed in earlier Table 1. For example, if the opaque type is Generic LSP Identifier, then the type-specific-key will be a uint32 value corresponding to the LSP. Please see the complete model for all other types.

Moreover, the binding tree defines only three types of sub-trees (i.e. lspid, src, and bidir) which is able to map the respective variants (vpn, recursive, and vpn-recursive) accordingly. For example, the key for opaque-type-src is [R, S, G, rd, recur-R, recur-RD], where basic type will specify (R, S,G,-, -, -), VPN type will specify (R, S,G, rd, -, -), recursive type will specify [R, S,G, -, recur-R, -] and VPN-recursive type will specify [R, S,G, -, recur-R, recur-rd].

It is important to take note of the following:

- o The address-family ipv4/ipv4 applies to "root" address in the mLDP binding tree. The other addresses (source, group, RP etc) do not have to be of the same address family type as the root.
- o The "recur-root-address" field applies to Recursive opaque type, and (recur-root-address, recur-rd) fields applies to VPN-Recursive opaque types as defined in [RFC6512]
- o In case of a recursive FEC, the address-family of the recur-root-address could be different than the address-family of the root address of original encapsulated MP FEC

The following diagram illustrates the FEC-label binding information structure for a P2MP (Transit IPv4 Source type) LSP on a branch/transit LSR:

```

FEC (root 2.2.2.2, S=192.168.1.1, G=224.1.1.1):
  type: p2mp
  upstream:
    advertised:
      peer 192.168.0.1:0, label 16000 (local)
  downstream:
    received:
      peer 192.168.0.2:0, label 17000 (remote)
      peer 192.168.0.3:0, label 18000 (remote)

```

Figure 19



The following diagram illustrates the FEC-label binding information structure for a similar MP2MP LSP on a branch/transit LSR:

```

FEC (root 2.2.2.2, RP=192.168.9.9, G=224.1.1.1):
  type: mp2mp
  upstream:
    advertised:
      peer 192.168.0.1:0, label 16000 (local)
    received:
      peer 192.168.0.1:0, label 17000 (remote)
  downstream:
    advertised:
      peer 192.168.0.2:0, label 16001 (local), MBB role=active
      peer 192.168.0.3:0, label 16002 (local), MBB role=standby
    received:
      peer 192.168.0.2:0, label 17001 (remote)
      peer 192.168.0.3:0, label 18001 (remote)

```

Figure 20

#### 4.3.1.3. Capabilities state

Like LDP, mLDP capabilities state comprise two types of information - global information and per-peer information.

#### 4.4. Notifications

mLDP notification module consists of notification related to changes in the operational state of an mLDP FEC. Following is a simplified graphical representation of the data model for mLDP notifications:

```

notifications:
  +---n mpls-ml dp-fec-event
    +---ro event-type?      oper-status-event-type
    +---ro tree-type?       multipoint-type
    +---ro root?            inet:ip-address
    +---ro (lsp-key-type)?
      +---:(lsp-id-based)
        | +---ro lsp-id?      uint16
      +---:(source-group-based)
        +---ro source-address? inet:ip-address
        +---ro group-address?  inet:ip-address

```

Figure 21

#### 4.5. Actions

Currently, no RPCs/actions are defined for mLDP.

#### 5. Open Items

Following is a list of open items that are to be discussed and addressed in future revisions of this document:

- o Close on augmentation off "mpls" list in "ietf-mpls" defined in [I-D.saad-mpls-base-yang]
- o Align operational state modeling with other routing procols and [I-D.openconfig-netmod-opstate]
- o Complete the section on Protocol-centric implementations and all-vrfs
- o Specify default values for configuration parameters
- o Revisit and cut down on the scope of the document and number of features it is trying to cover
- o Split the model into a base and extended items
- o Add statistics for mLDP root LSPs and bindings
- o Extend the "Configured Leaf LSPs" for various type of opaque-types
- o Extend mLDP notifications for other types of opaque values as well
- o Close on single vs separate document for mLDP Yang

#### 6. YANG Specification

Following are actual YANG definition for LDP and mLDP constructs defined earlier in the document.

```
<CODE BEGINS> file "ietf-mpls-ldp@2016-07-08.yang" -->

module ietf-mpls-ldp {
  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-ldp";
  // replace with IANA namespace when assigned
  prefix ldp;

  import ietf-inet-types {
```

```
    prefix "inet";
  }

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

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-ip {
    prefix "ip";
  }

  import ietf-routing {
    prefix "rt";
  }

  import ietf-mpls {
    prefix "mpls";
  }

  organization
    "IETF MPLS Working Group";
  contact
    "WG Web:      <http://tools.ietf.org/wg/teas/>
    WG List:      <mailto:teas@ietf.org>

    WG Chair: Loa Andersson
               <mailto:loa@pi.nu>

    WG Chair: Ross Callon
               <mailto:rcallon@juniper.net>

    WG Chair: George Swallow
               <mailto:swallow.ietf@gmail.com>

    Editor: Kamran Raza
            <mailto:skraza@cisco.com>

    Editor: Rajiv Asati
            <mailto:rajiva@cisco.com>

    Editor: Xufeng Liu
            <mailto:xliu@kuatrotech.com>

    Editor: Santosh Esale
```

<mailto:sesale@juniper.net>

Editor: Xia Chen  
<mailto:jescia.chenxia@huawei.com>

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

description

"This YANG module defines the essential components for the management of Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).";

revision 2016-07-08 {

description

"Initial revision.";

reference

"RFC XXXX: YANG Data Model for MPLS LDP and mLDP.";

}

/\*

\* Features

\*/

feature admin-down-config {

description

"This feature indicates that the system allows to configure administrative down on a VRF instance and a peer.";

}

feature all-af-policy-config {

description

"This feature indicates that the system allows to configure policies that are applied to all address families.";

}

feature capability-end-of-lib {

description

"This feature indicates that the system allows to configure LDP end-of-lib capability.";

}

feature capability-ml dp-hsmp {

description

"This feature indicates that the system allows to configure mLDP hub-and-spoke-multipoint capability.";

}

```
feature capability-mldp-node-protection {
  description
    "This feature indicates that the system allows to configure
    mLDP node-protection capability.";
}

feature capability-typed-wildcard-fec {
  description
    "This feature indicates that the system allows to configure
    LDP typed-wildcard-fec capability.";
}

feature capability-upstream-label-assignment {
  description
    "This feature indicates that the system allows to configure
    LDP upstream label assignment capability.";
}

feature forwarding-nexthop-config {
  description
    "This feature indicates that the system allows to configure
    forwarding nexthop on interfaces.";
}

feature global-session-authentication {
  description
    "This feature indicates that the system allows to configure
    authentication at global level.";
}

feature graceful-restart-helper-mode {
  description
    "This feature indicates that the system supports graceful
    restart helper mode.";
}

feature mldp {
  description
    "This feature indicates that the system supports Multicast
    LDP (mLDP).";
}

feature mldp-mofrr {
  description
    "This feature indicates that the system supports mLDP
    Multicast only FRR (MoFRR).";
}
```

```
feature per-interface-timer-config {
  description
    "This feature indicates that the system allows to configure
    interface hello timers at the per-interface level.";
}

feature per-peer-graceful-restart-config {
  description
    "This feature indicates that the system allows to configure
    graceful restart at the per-peer level.";
}

feature per-peer-session-attributes-config {
  description
    "This feature indicates that the system allows to configure
    session attributes at the per-peer level.";
}

feature policy-extended-discovery-config {
  description
    "This feature indicates that the system allows to configure
    policies to control the acceptance of extended neighbor
    discovery hello messages.";
}

feature policy-label-assignment-config {
  description
    "This feature indicates that the system allows to configure
    policies to assign labels according to certain prefixes.";
}

feature policy-ordered-label-config {
  description
    "This feature indicates that the system allows to configure
    ordered label policies.";
}

feature session-downstream-on-demand-config {
  description
    "This feature indicates that the system allows to configure
    session downstream-on-demand";
}

/*
 * Typedefs
 */
typedef ldp-address-family {
  type identityref {
```

```
    base rt:address-family;
  }
  description
    "LDP address family type.";
}

typedef duration32-inf {
  type union {
    type uint32;
    type enumeration {
      enum "infinite" {
        description "The duration is infinite.";
      }
    }
  }
  units seconds;
  description
    "Duration represented as 32 bit seconds with infinite.";
}

typedef advertised-received {
  type enumeration {
    enum advertised {
      description "Advertised information.";
    }
    enum received {
      description "Received information.";
    }
  }
  description
    "Received or advertised.";
}

typedef downstream-upstream {
  type enumeration {
    enum downstream {
      description "Downstream information.";
    }
    enum upstream {
      description "Upstream information.";
    }
  }
  description
    "Received or advertised.";
}

typedef label-adv-mode {
  type enumeration {
```

```
    enum downstream-unsolicited {
      description "Downstream Unsolicited.";
    }
    enum downstream-on-demand {
      description "Downstream on Demand.";
    }
  }
  description
    "Label Advertisement Mode.";
}

typedef mpls-interface-ref {
  type leafref {
    path "/rt:routing/mpls:mpls/mpls:interface/mpls:name";
  }
  description
    "This type is used by data models that need to reference
    mpls interfaces.";
}

typedef multipoint-type {
  type enumeration {
    enum p2mp {
      description "Point to multipoint.";
    }
    enum mp2mp {
      description "Multipoint to multipoint.";
    }
  }
  description
    "p2mp or mp2mp.";
}

typedef neighbor-list-ref {
  type string;
  description
    "A type for a reference to a neighbor list.";
}

typedef peer-list-ref {
  type string;
  description
    "A type for a reference to a peer list.";
}

typedef prefix-list-ref {
  type string;
  description
```



```
    "A type for a reference to a prefix list.";
}

typedef oper-status-event-type {
  type enumeration {
    enum up {
      value 1;
      description
        "Operational status changed to up.";
    }
    enum down {
      value 2;
      description
        "Operational status changed to down.";
    }
  }
  description "Operational status event type for notifications.";
}

typedef route-distinguisher {
  type string {
  }
  description
    "Type definition for route distinguisher.";
  reference
    "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
}

/*
 * Identities
 */
identity adjacency-flag-base {
  description "Base type for adjacency flags.";
}

identity adjacency-flag-active {
  base "adjacency-flag-base";
  description
    "This adjacency is configured and actively created.";
}

identity adjacency-flag-passive {
  base "adjacency-flag-base";
  description
    "This adjacency is not configured and passively accepted.";
}

/*
```

```
* Groupings
*/

grouping adjacency-state-attributes {
  description
    "Adjacency state attributes.";

  leaf-list flag {
    type identityref {
      base "adjacency-flag-base";
    }
    description "Adjacency flags.";
  }
  container hello-holdtime {
    description "Hello holdtime state.";
    leaf adjacent {
      type uint16;
      units seconds;
      description "Peer holdtime.";
    }
    leaf negotiated {
      type uint16;
      units seconds;
      description "Negotiated holdtime.";
    }
    leaf remaining {
      type uint16;
      units seconds;
      description "Remaining holdtime.";
    }
  }

  leaf next-hello {
    type uint16;
    units seconds;
    description "Time to send the next hello message.";
  }

  container statistics {
    description
      "Statistics objects.";

    leaf discontinuity-time {
      type yang:date-and-time;
      mandatory true;
      description
        "The time on the most recent occasion at which any one or
        more of this interface's counters suffered a
```

```
        discontinuity.  If no such discontinuities have occurred
        since the last re-initialization of the local management
        subsystem, then this node contains the time the local
        management subsystem re-initialized itself.";
    }

    leaf hello-received {
        type yang:counter64;
        description
            "The number of hello messages received.";
    }
    leaf hello-dropped {
        type yang:counter64;
        description
            "The number of hello messages received.";
    }
} // statistics
} // adjacency-state-attributes

grouping basic-discovery-timers {
    description
        "Basic discovery timer attributes.";
    leaf hello-holdtime {
        type uint16 {
            range 15..3600;
        }
        units seconds;
        description
            "The time interval for which a LDP link Hello adjacency
            is maintained in the absence of link Hello messages from
            the LDP neighbor";
    }
    leaf hello-interval {
        type uint16 {
            range 5..1200;
        }
        units seconds;
        description
            "The interval between consecutive LDP link Hello messages
            used in basic LDP discovery";
    }
} // basic-discovery-timers

grouping binding-address-state-attributes {
    description
        "Address binding attributes";
    leaf advertisement-type {
        type advertised-received;
```

```

        description
            "Received or advertised.";
    }
    leaf peer {
        type leafref {
            path "../.../.../.../.../peers/peer/lsr-id";
        }
        must "../advertisement-type = 'received'" {
            description
                "Applicable for received address.";
        }
        description
            "LDP peer from which this address is received.";
    } // peer
} // binding-address-state-attributes

grouping binding-label-state-attributes {
    description
        "Label binding attributes";
    list peer {
        key "peer advertisement-type";
        description
            "List of advertised and received peers.";
        leaf peer {
            type leafref {
                path "../.../.../.../.../.../peers/peer/lsr-id";
            }
            description
                "LDP peer from which this binding is received,
                or to which this binding is advertised.";
        }
        leaf advertisement-type {
            type advertised-received;
            description
                "Received or advertised.";
        }
        leaf label {
            type mpls:mpls-label;
            description
                "Advertised (outbound) or received (inbound)
                label.";
        }
        leaf used-in-forwarding {
            type boolean;
            description
                "'true' if the lable is used in forwarding.";
        }
    } // peer
}

```

```
} // binding-label-state-attributes

grouping extended-discovery-policy-attributes {
  description
    "LDP policy to control the acceptance of extended neighbor
    discovery hello messages.";
  container hello-accept {
    if-feature policy-extended-discovery-config;
    description
      "Extended discovery acceptance policies.";

    leaf enable {
      type boolean;
      description
        "'true' to accept; 'false' to deny.";
    }
    leaf neighbor-list {
      type neighbor-list-ref;
      description
        "The name of a peer ACL.";
    }
  } // hello-accept
} // extended-discovery-policy-attributes

grouping extended-discovery-timers {
  description
    "Extended discovery timer attributes.";
  leaf hello-holdtime {
    type uint16 {
      range 15..3600;
    }
    units seconds;
    description
      "The time interval for which LDP targeted Hello adjacency
      is maintained in the absence of targeted Hello messages
      from an LDP neighbor.";
  }
  leaf hello-interval {
    type uint16 {
      range 5..3600;
    }
    units seconds;
    description
      "The interval between consecutive LDP targeted Hello
      messages used in extended LDP discovery.";
  }
}
```

```
} // extended-discovery-timers

grouping global-attributes {
  description "Configuration attributes at global level.";

  uses instance-attributes;
} // global-attributes

grouping graceful-restart-attributes {
  description
    "Graceful restart configuration attributes.";
  container graceful-restart {
    description
      "Attributes for graceful restart.";
    leaf enable {
      type boolean;
      description
        "Enable or disable graceful restart.";
    }
    leaf helper-enable {
      if-feature graceful-restart-helper-mode;
      type boolean;
      description
        "Enable or disable graceful restart helper mode.";
    }
    leaf reconnect-time {
      type uint16 {
        range 10..1800;
      }
      units seconds;
      description
        "Specifies the time interval that the remote LDP peer
        must wait for the local LDP peer to reconnect after the
        remote peer detects the LDP communication failure.";
    }
    leaf recovery-time {
      type uint16 {
        range 30..3600;
      }
      units seconds;
      description
        "Specifies the time interval, in seconds, that the remote
        LDP peer preserves its MPLS forwarding state after
        receiving the Initialization message from the restarted
        local LDP peer.";
    }
    leaf forwarding-holdtime {
      type uint16 {
```

```
        range 30..3600;
    }
    units seconds;
    description
        "Specifies the time interval, in seconds, before the
        termination of the recovery phase.";
    }
} // graceful-restart
} // graceful-restart-attributes

grouping graceful-restart-attributes-per-peer {
    description
        "Per peer graceful restart configuration attributes.";
    container graceful-restart {
        description
            "Attributes for graceful restart.";
        leaf enable {
            type boolean;
            description
                "Enable or disable graceful restart.";
        }
        leaf reconnect-time {
            type uint16 {
                range 10..1800;
            }
            units seconds;
            description
                "Specifies the time interval that the remote LDP peer
                must wait for the local LDP peer to reconnect after the
                remote peer detects the LDP communication failure.";
        }
        leaf recovery-time {
            type uint16 {
                range 30..3600;
            }
            units seconds;
            description
                "Specifies the time interval, in seconds, that the remote
                LDP peer preserves its MPLS forwarding state after
                receiving the Initialization message from the restarted
                local LDP peer.";
        }
    }
} // graceful-restart
} // graceful-restart-attributes-per-peer

grouping instance-attributes {
    description "Configuration attributes at instance level.";
```

```
container capability {
  description "Configure capability.";
  container end-of-lib {
    if-feature capability-end-of-lib;
    description
      "Configure end-of-lib capability.";
    leaf enable {
      type boolean;
      description
        "Enable end-of-lib capability.";
    }
  }
  container typed-wildcard-fec {
    if-feature capability-typed-wildcard-fec;
    description
      "Configure typed-wildcard-fec capability.";
    leaf enable {
      type boolean;
      description
        "Enable typed-wildcard-fec capability.";
    }
  }
  container upstream-label-assignment {
    if-feature capability-upstream-label-assignment;
    description
      "Configure upstream label assignment capability.";
    leaf enable {
      type boolean;
      description
        "Enable upstream label assignment.";
    }
  }
  container mldp {
    if-feature mldp;

    description
      "Multipoint capabilities.";
    uses mldp-capabilities;
  }
} // capability

uses graceful-restart-attributes;

leaf igp-synchronization-delay {
  type uint16 {
    range 3..60;
  }
  units seconds;
}
```



```
    description
      "Sets the interval that the LDP waits before notifying the
       Interior Gateway Protocol (IGP) that label exchange is
       completed so that IGP can start advertising the normal
       metric for the link.";
  }
  leaf lsr-id {
    type yang:dotted-quad;
    description "Router ID.";
  }
} // instance-attributes

grouping ldp-adjacency-ref {
  description
    "An absolute reference to an LDP adjacency.";
  choice hello-adjacency-type {
    description
      "Interface or targeted adjacency.";
    case targeted {
      container targeted {
        description "Targeted adjacency.";
        leaf target-address {
          type inet:ip-address;
          description
            "The target address.";
        }
      } // targeted
    }
    case link {
      container link {
        description "Link adjacency.";
        leaf next-hop-interface {
          type mpls-interface-ref;
          description
            "Interface connecting to next-hop.";
        }
        leaf next-hop-address {
          type inet:ip-address;
          must "../next-hop-interface" {
            description
              "Applicable when interface is specified.";
          }
        }
        description
          "IP address of next-hop.";
      }
    } // link
  }
}
```

```
    }  
  } // ldp-adjacency-ref  
  
  grouping ldp-fec-event {  
    description  
      "A LDP FEC event.";   
    leaf prefix {  
      type inet:ip-prefix;  
      description  
        "FEC.";   
    }  
  } // ldp-fec-event  
  
  grouping ldp-peer-ref {  
    description  
      "An absolute reference to an LDP peer.";   
    leaf peer-ref {  
      type leafref {  
        path "/rt:routing/rt:control-plane-protocols/mpls-ldp/"  
          + "peers/peer/lsr-id";  
      }  
      description  
        "Reference to an LDP peer.";   
    }  
  } // ldp-peer-ref  
  
  grouping mldp-capabilities {  
    description  
      "mLDP capabilities.";   
    container p2mp {  
      description  
        "Configure point-to-multipoint capability.";   
      leaf enable {  
        type boolean;  
        description  
          "Enable point-to-multipoint.";   
      }  
    }  
    container mp2mp {  
      description  
        "Configure multipoint-to-multipoint capability.";   
      leaf enable {  
        type boolean;  
        description  
          "Enable multipoint-to-multipoint.";   
      }  
    }  
    container make-before-break {
```

```
description
  "Configure make-before-break capability.";
leaf enable {
  type boolean;
  description
    "Enable make-before-break.";
}
leaf switchover-delay {
  type uint16;
  units seconds;
  description
    "Switchover delay in seconds.";
}
leaf timeout {
  type uint16;
  units seconds;
  description
    "Timeout in seconds.";
}
}
container hub-and-spoke {
  if-feature capability-mldp-hsmp;
  description
    "Configure hub-and-spoke-multipoint capability.";
  reference
    "RFC7140: LDP Extensions for Hub and Spoke Multipoint
    Label Switched Path";
  leaf enable {
    type boolean;
    description
      "Enable hub-and-spoke-multipoint.";
  }
}
container node-protection {
  if-feature capability-mldp-node-protection;
  description
    "Configure node-protection capability.";
  reference
    "RFC7715: mLDP Node Protection.";
  leaf plr {
    type boolean;
    description
      "Point of Local Repair capable for MP LSP node
      protection.";
  }
}
container merge-point {
  description
    "Merge Point capable for MP LSP node protection.";
```

```
    leaf enable {
      type boolean;
      description
        "Enable merge point capability.";
    }
    leaf targeted-session-teardown-delay {
      type uint16;
      units seconds;
      description
        "Targeted session teardown delay.";
    }
  } // merge-point
} // mldp-capabilities

grouping mldp-configured-lsp-roots {
  description
    "mLDP roots containers.";

  container roots-ipv4 {

    when "../../af = 'ipv4'" {
      description
        "Only for IPv4.";
    }
    description
      "Configured IPv4 multicast LSPs.";
    list root {
      key "root-address";
      description
        "List of roots for configured multicast LSPs.";

      leaf root-address {
        type inet:ipv4-address;
        description
          "Root address.";
      }
    }

    list lsp {
      must "(lsp-id = 0 and source-address != '0.0.0.0' and "
        + "group-address != '0.0.0.0') or "
        + "(lsp-id != 0 and source-address = '0.0.0.0' and "
        + "group-address = '0.0.0.0')";
      description
        "A LSP can be identified by either <lsp-id> or
        <source-address, group-address>.";
    }
    key "lsp-id source-address group-address";
  }
}
```

```
    description
      "List of LSPs.";
    leaf lsp-id {
      type uint16;
      description "ID to identify the LSP.";
    }
    leaf source-address {
      type inet:ipv4-address;
      description
        "Source address.";
    }
    leaf group-address {
      type inet:ipv4-address-no-zone;
      description
        "Group address.";
    }
  } // list lsp
} // list root
} // roots-ipv4

container roots-ipv6 {

  when "../../../af = 'ipv6'" {
    description
      "Only for IPv6.";
  }
  description
    "Configured IPv6 multicast LSPs.";

  list root {
    key "root-address";
    description
      "List of roots for configured multicast LSPs.";

    leaf root-address {
      type inet:ipv6-address;
      description
        "Root address.";
    }
  }

  list lsp {
    must "(lsp-id = 0 and source-address != '::' and "
      + "group-address != '::') or "
      + "(lsp-id != 0 and source-address = '::' and "
      + "group-address = '::'))" {
      description
        "A LSP can be identified by either <lsp-id> or
        <source-address, group-address>.";
    }
  }
}
```

```
    }
    key "lsp-id source-address group-address";
    description
      "List of LSPs.";
    leaf lsp-id {
      type uint16;
      description "ID to identify the LSP.";
    }
    leaf source-address {
      type inet:ipv6-address;
      description
        "Source address.";
    }
    leaf group-address {
      type inet:ipv6-address-no-zone;
      description
        "Group address.";
    }
  } // list lsp
} // list root
} // roots-ipv6
} // mldp-configured-lsp-roots

grouping mldp-fec-event {
  description
    "A mLDP FEC event.";
  leaf tree-type {
    type multipoint-type;
    description
      "p2mp or mp2mp.";
  }
  leaf root {
    type inet:ip-address;
    description
      "Root address.";
  }
  choice lsp-key-type {
    description
      "LSP ID based or source-group based .";
    case lsp-id-based {
      leaf lsp-id {
        type uint16;
        description
          "ID to identify the LSP.";
      }
    }
    case source-group-based {
      leaf source-address {

```

```
        type inet:ip-address;
        description

            "LSP source address.";
    }
    leaf group-address {
        type inet:ip-address;
        description
            "Multicast group address.";
    }
} // case source-group-based
}
} // mldp-fec-event

grouping mldp-binding-label-state-attributes {
    description
        "mLDP label binding attributes.";

    leaf multipoint-type {
        type multipoint-type;
        description
            "The type of mutipoint, p2mp or mp2mp.";
    }
    list peer {
        key "direction peer advertisement-type";
        description
            "List of advertised and received peers.";
        leaf direction {
            type downstream-upstream;
            description
                "Downstream or upstream.";
        }
        leaf peer {
            type leafref {
                path
                    ".../.../.../.../.../.../.../.../.../peers/peer/lsr-id";
            }
            description
                "LDP peer from which this binding is received,
                or to which this binding is advertised.";
        }
        leaf advertisement-type {
            type advertised-received;
            description
                "Advertised or received.";
        }
        leaf label {
            type mpls:mpls-label;
        }
    }
}
```

```
        description
            "Advertised (outbound) or received (inbound) label.";
    }
    leaf mbb-role {
        when "../direction = 'upstream'" {
            description
                "For upstream.";
        }
        type enumeration {
            enum none {
                description "MBB is not enabled.";
            }
            enum active {
                description "This LSP is active.";
            }
            enum inactive {
                description "This LSP is inactive.";
            }
        }
        description
            "The MBB status of this LSP.";
    }
    leaf mofrr-role {
        when "../direction = 'upstream'" {
            description
                "For upstream.";
        }
        type enumeration {
            enum none {
                description "MOFRR is not enabled.";
            }
            enum primary {
                description "This LSP is primary.";
            }
            enum backup {
                description "This LSP is backup.";
            }
        }
        description
            "The MOFRR status of this LSP.";
    }
} // peer
} // mldp-binding-label-state-attributes

grouping peer-af-policy-container {
    description
        "LDP policy attribute container under peer address-family.";
    container label-policy {
```



```
description
  "Label policy attributes.";
container advertise {
  description
    "Label advertising policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Applies the prefix list to outgoing label
      advertisements.";
  }
}
container accept {
  description
    "Label advertisement acceptance policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Applies the prefix list to incoming label
      advertisements.";
  }
} // accept
} // label-policy
} // peer-af-policy-container

grouping peer-attributes {
  description "Peer configuration attributes.";

  leaf session-ka-holdtime {
    type uint16 {
      range 45..3600;
    }
    units seconds;
    description
      "The time interval after which an inactive LDP session
      terminates and the corresponding TCP session closes.
      Inactivity is defined as not receiving LDP packets from the
      peer.";
  }
  leaf session-ka-interval {
    type uint16 {
      range 15..1200;
    }
    units seconds;
    description
      "The interval between successive transmissions of keepalive
      packets. Keepalive packets are only sent in the absence of
      other LDP packets transmitted over the LDP session.";
  }
}
```

```
    }  
  } // peer-attributes  
  
  grouping peer-authentication {  
    description  
      "Peer authentication attributes.";  
    leaf session-authentication-md5-password {  
      type string {  
        length "1..80";  
      }  
      description  
        "Assigns an encrypted MD5 password to an LDP  
        peer";  
    } // md5-password  
  } // peer-authentication  
  
  grouping peer-state-derived {  
    description "Peer derived state attributes.";  
  
    container label-advertisement-mode {  
      description "Label advertisement mode state.";  
      leaf local {  
        type label-adv-mode;  
        description  
          "Local Label Advertisement Mode.";  
      }  
      leaf peer {  
        type label-adv-mode;  
        description  
          "Peer Label Advertisement Mode.";  
      }  
      leaf negotiated {  
        type label-adv-mode;  
        description  
          "Negotiated Label Advertisement Mode.";  
      }  
    }  
  }  
  leaf next-keep-alive {  
    type uint16;  
    units seconds;  
    description "Time to send the next KeepAlive message.";  
  }  
  
  leaf peer-ldp-id {  
    type yang:dotted-quad;  
    description "Peer LDP ID.";  
  }  
}
```

```
container received-peer-state {
  description "Peer features.";

  uses graceful-restart-attributes-per-peer;

  container capability {
    description "Configure capability.";
    container end-of-lib {
      description
        "Configure end-of-lib capability.";
      leaf enable {
        type boolean;
        description
          "Enable end-of-lib capability.";
      }
    }
  }
  container typed-wildcard-fec {
    description
      "Configure typed-wildcard-fec capability.";
    leaf enable {
      type boolean;
      description
        "Enable typed-wildcard-fec capability.";
    }
  }
  container upstream-label-assignment {
    description
      "Configure upstream label assignment capability.";
    leaf enable {
      type boolean;
      description
        "Enable upstream label assignment.";
    }
  }
  container mldp {
    if-feature mldp;
    description
      "Multipoint capabilities.";

    container p2mp {
      description
        "Configure point-to-multipoint capability.";
      leaf enable {
        type boolean;
        description
          "Enable point-to-multipoint.";
      }
    }
  }
}
```

```
container mp2mp {
  description
    "Configure multipoint-to-multipoint capability.";
  leaf enable {
    type boolean;
    description
      "Enable multipoint-to-multipoint.";
  }
}
container make-before-break {
  description
    "Configure make-before-break capability.";
  leaf enable {
    type boolean;
    description
      "Enable make-before-break.";
  }
}
container hub-and-spoke {
  description
    "Configure hub-and-spoke-multipoint capability.";
  reference
    "RFC7140: LDP Extensions for Hub and Spoke Multipoint
    Label Switched Path";
  leaf enable {
    type boolean;
    description
      "Enable hub-and-spoke-multipoint.";
  }
}
container node-protection {
  description
    "Configure node-protection capability.";
  reference
    "RFC7715: mLDP Node Protection.";
  leaf plr {
    type boolean;
    description
      "Point of Local Repair capable for MP LSP node
      protection.";
  }
  leaf merge-point {
    type boolean;
    description
      "Merge Point capable for MP LSP node protection.";
  } // merge-point
} // node-protection
} // mldp
```

```
    } // capability
  } // received-peer-state

  container session-holdtime {
    description "Session holdtime state.";
    leaf peer {
      type uint16;
      units seconds;
      description "Peer holdtime.";
    }
    leaf negotiated {
      type uint16;
      units seconds;
      description "Negotiated holdtime.";
    }
    leaf remaining {
      type uint16;
      units seconds;
      description "Remaining holdtime.";
    }
  } // session-holdtime

  leaf session-state {
    type enumeration {
      enum non-existent {
        description "NON EXISTENT state. Transport disconnected.";
      }
      enum initialized {
        description "INITIALIZED state.";
      }
      enum openrec {
        description "OPENREC state.";
      }
      enum opensent {
        description "OPENSENT state.";
      }
      enum operational {
        description "OPERATIONAL state.";
      }
    }
    description
      "Representing the operational status.";
  }

  container tcp-connection {
    description "TCP connection state.";
    leaf local-address {
      type inet:ip-address;
    }
  }
}
```

```
        description "Local address.";
    }
    leaf local-port {
        type inet:port-number;
        description "Local port.";
    }
    leaf remote-address {
        type inet:ip-address;
        description "Remote address.";
    }
    leaf remote-port {
        type inet:port-number;
        description "Remote port.";
    }
} // tcp-connection

leaf up-time {
    type string;
    description "Up time. The interval format in ISO 8601.";
}

container statistics {
    description
        "Statistics objects.";

    leaf discontinuity-time {
        type yang:date-and-time;
        mandatory true;
        description
            "The time on the most recent occasion at which any one or
            more of this interface's counters suffered a
            discontinuity. If no such discontinuities have occurred
            since the last re-initialization of the local management
            subsystem, then this node contains the time the local
            management subsystem re-initialized itself.";
    }

    container received {
        description "Inbound statistics.";
        uses statistics-peer-received-sent;
    }
    container sent {
        description "Outbound statistics.";
        uses statistics-peer-received-sent;
    }
}

leaf total-addresses {
    type uint32;
```

```
        description
            "The number of learned addresses.";
    }
    leaf total-labels {
        type uint32;
        description
            "The number of learned labels.";
    }
    leaf total-fec-label-bindings {
        type uint32;
        description
            "The number of learned label-address bindings.";
    }
} // statistics
} // peer-state-derived

grouping policy-container {
    description
        "LDP policy attributes.";
    container label-policy {
        description
            "Label policy attributes.";
        container independent-mode {
            description
                "Independent label policy attributes.";
            container assign {

                if-feature policy-label-assignment-config;
                description
                    "Label assignment policies";
                choice prefix-option {
                    description
                        "Use either prefix-list or host-routes-only.";
                    case prefix-list {
                        leaf prefix-list {
                            type prefix-list-ref;
                            description
                                "Assign labels according to certain prefixes.";
                        }
                    }
                    case host-routes-only {
                        leaf host-routes-only {
                            type boolean;
                            description
                                "'true' to apply host routes only.";
                        }
                    }
                }
            } // prefix-option
        }
    }
}
```

```
}
container advertise {
  description
    "Label advertising policies.";
  container explicit-null {
    description
      "Enables an egress router to advertise an
       explicit null label (value 0) in place of an
       implicit null label (value 3) to the
       penultimate hop router.";
    leaf enable {
      type boolean;
      description
        "'true' to enable explicit null.";
    }
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Prefix list name. Applies the filters in the
         specified prefix list to label
         advertisements.
         If the prefix list is not specified, explicit
         null label advertisement is enabled for all
         directly connected prefixes.";
    }
  }
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Applies the prefix list to outgoing label
       advertisements.";
  }
}
container accept {
  description
    "Label advertisement acceptance policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Applies the prefix list to incoming label
       advertisements.";
  }
}
} // independent-mode
container ordered-mode {
  if-feature policy-ordered-label-config;
  description
```



```
    "Ordered label policy attributes.";
  container egress-lsr {
    description
      "Egress LSR label assignment policies";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Assign labels according to certain prefixes.";
    }
  }
  container advertise {
    description
      "Label advertising policies.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Applies the prefix list to outgoing label
        advertisements.";
    }
  }
  container accept {
    description
      "Label advertisement acceptance policies.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Applies the prefix list to incoming label
        advertisements.";
    }
  }
} // ordered-mode
} // label-policy
} // policy-container

grouping statistics-peer-received-sent {
  description
    "Inbound and outbound statistic counters.";
  leaf total-octets {
    type yang:counter64;
    description
      "The total number of octets sent or received.";
  }
  leaf total-messages {
    type yang:counter64;
    description
      "The number of messages sent or received.";
  }
  leaf address {
```

```
    type yang:counter64;
    description
      "The number of address messages sent or received.";
  }
  leaf address-withdraw {
    type yang:counter64;
    description
      "The number of address-withdraw messages sent or received.";
  }
  leaf initialization {
    type yang:counter64;
    description
      "The number of initialization messages sent or received.";
  }
  leaf keepalive {
    type yang:counter64;
    description
      "The number of keepalive messages sent or received.";
  }
  leaf label-abort-request {
    type yang:counter64;
    description
      "The number of label-abort-request messages sent or
      received.";
  }
  leaf label-mapping {
    type yang:counter64;
    description
      "The number of label-mapping messages sent or received.";
  }
  leaf label-release {
    type yang:counter64;
    description
      "The number of label-release messages sent or received.";
  }
  leaf label-request {
    type yang:counter64;
    description
      "The number of label-request messages sent or received.";
  }
  leaf label-withdraw {
    type yang:counter64;
    description
      "The number of label-withdraw messages sent or received.";
  }
  leaf notification {
    type yang:counter64;
    description
```

```
        "The number of messages sent or received.";
    }
} // statistics-peer-received-sent

/*
 * Configuration data nodes
 */

augment "/rt:routing/rt:control-plane-protocols" {
    description "LDP augmentation.";

    container mpls-ldp {
        presence "Container for LDP protocol.";
        description
            "Container for LDP protocol.";

        container global {
            description
                "Global attributes for LDP.";
            container config {
                description
                    "Configuration data.";
                uses global-attributes;
            }
            container state {
                config false;
                description
                    "Operational state data.";
                uses global-attributes;
            }
        }

        container mldp {
            if-feature mldp;
            description
                "mLDP attributes at per instance level. Defining
                attributes here does not enable any MP capabilities.
                MP capabilities need to be explicitly enabled under
                container capability.";

            container config {
                description
                    "Configuration data.";
                leaf enable {
                    type boolean;
                    description
                        "Enable mLDP.";
                }
            }
        }
    }
}
```

```
container state {
  config false;
  description

    "Operational state data.";
  leaf enable {
    type boolean;
    description
      "Enable mLDP.";
  }
}

list address-family {
  key "afi";
  description
    "Per-af params.";
  leaf afi {
    type ldp-address-family;
    description
      "Address family type value.";
  }
}

container config {
  description
    "Configuration data.";
  container multicast-only-frr {
    if-feature mldp-mofrr;
    description
      "Multicast only FRR (MoFRR) policy.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Enables MoFRR for the specified access list.";
    }
  } // multicast-only-frr
  container recursive-fec {
    description
      "Recursive FEC policy.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Enables recursive FEC for the specified access
        list.";
    }
  } // recursive-for
}
container state {
  config false;
```

```
description
  "Operational state data.";
container multicast-only-frr {
  if-feature mldp-mofrr;

  description
    "Multicast only FRR (MoFRR) policy.";
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Enables MoFRR for the specified access list.";
  }
} // multicast-only-frr
container recursive-fec {
  description
    "Recursive FEC policy.";
  leaf prefix-list {
    type prefix-list-ref;
    description
      "Enables recursive FEC for the specified access
      list.";
  }
} // recursive-fec

container ipv4 {
  when "../..afi = 'ipv4'" {
    description
      "Only for IPv4.";
  }
  description
    "IPv4 state information.";
  container roots {
    description
      "IPv4 multicast LSP roots.";
    list root {
      key "root-address";
      description
        "List of roots for configured multicast LSPs.";

      leaf root-address {
        type inet:ipv4-address;
        description
          "Root address.";
      }

      leaf is-self {
        type boolean;
        description
```

```
        "This is the root.";
    }

    list reachability {
        key "address interface";
        description
            "A next hop for reachability to root,
            as a RIB view.";
        leaf address {
            type inet:ipv4-address;
            description
                "The next hop address to reach root.";
        }
        leaf interface {
            type mpls-interface-ref;
            description
                "Interface connecting to next-hop.";
        }
        leaf peer {
            type leafref {
                path
                    "../.../peers/peer/"
                    + "lsr-id";
            }
            description
                "LDP peer from which this next hop can be
                reached.";
        }
    }
} // list root
} // roots
container bindings {
    description
        "mLDP FEC to label bindings.";
    container opaque-type-lspid {
        description
            "The type of opaque value element is
            the generic LSP identifier";
        reference
            "RFC6388: Label Distribution Protocol
            Extensions for Point-to-Multipoint and
            Multipoint-to-Multipoint Label Switched
            Paths.";
        list fec-label {
            key
                "root-address lsp-id "
                + "recur-root-address recur-rd";
            description
```

```
    "List of FEC to label bindings.";
  leaf root-address {
    type inet:ipv4-address;
    description
      "Root address.";
  }
  leaf lsp-id {
    type uint32;
    description "ID to identify the LSP.";
  }
  leaf recur-root-address {
    type inet:ip-address;
    description
      "Recursive root address.";
    reference
      "RFC6512: Using Multipoint LDP When the
      Backbone Has No Route to the Root";
  }
  leaf recur-rd {
    type route-distinguisher;
    description
      "Route Distinguisher in the VPN-Recursive
      Opaque Value.";
    reference
      "RFC6512: Using Multipoint LDP When the
      Backbone Has No Route to the Root";
  }
  uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-lspid

container opaque-type-src {
  description
    "The type of opaque value element is
    the transit source TLV";
  reference
    "RFC6826: Multipoint LDP In-Band Signaling for
    Point-to-Multipoint and
    Multipoint-to-Multipoint Label Switched
    Paths.";
  list fec-label {
    key
      "root-address source-address group-address "
      + "rd recur-root-address recur-rd";
    description
      "List of FEC to label bindings.";
    leaf root-address {
      type inet:ipv4-address;
```

```
        description
            "Root address.";
    }
    leaf source-address {
        type inet:ip-address;
        description
            "Source address.";
    }
    leaf group-address {
        type inet:ip-address-no-zone;
        description
            "Group address.";
    }
    leaf rd {
        type route-distinguisher;
        description
            "Route Distinguisher.";
        reference
            "RFC7246: Multipoint Label Distribution
            Protocol In-Band Signaling in a Virtual
            Routing and Forwarding (VRF) Table
            Context.";
    }
    leaf recur-root-address {
        type inet:ip-address;
        description
            "Recursive root address.";
        reference
            "RFC6512: Using Multipoint LDP When the
            Backbone Has No Route to the Root";
    }
    leaf recur-rd {
        type route-distinguisher;
        description
            "Route Distinguisher in the VPN-Recursive
            Opaque Value.";
        reference
            "RFC6512: Using Multipoint LDP When the
            Backbone Has No Route to the Root";
    }
    uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-src

container opaque-type-bidir {
    description
        "The type of opaque value element is
        the generic LSP identifier";
```



```
reference
  "RFC6826: Multipoint LDP In-Band Signaling for
  Point-to-Multipoint and
  Multipoint-to-Multipoint Label Switched
  Paths.";
list fec-label {
  key
    "root-address rp group-address "
    + "rd recur-root-address recur-rd";
  description
    "List of FEC to label bindings.";
  leaf root-address {
    type inet:ipv4-address;
    description
      "Root address.";
  }
  leaf rp {
    type inet:ip-address;
    description
      "RP address.";
  }
  leaf group-address {
    type inet:ip-address-no-zone;
    description
      "Group address.";
  }
  leaf rd {
    type route-distinguisher;
    description
      "Route Distinguisher.";
    reference
      "RFC7246: Multipoint Label Distribution
      Protocol In-Band Signaling in a Virtual
      Routing and Forwarding (VRF) Table
      Context.";
  }
  leaf recur-root-address {
    type inet:ip-address;
    description
      "Recursive root address.";
    reference
      "RFC6512: Using Multipoint LDP When the
      Backbone Has No Route to the Root";
  }
  leaf recur-rd {
    type route-distinguisher;
    description
      "Route Distinguisher in the VPN-Recursive
```

```
        Opaque Value.";
        reference
        "RFC6512: Using Multipoint LDP When the
        Backbone Has No Route to the Root";
    }
    uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-bidir
} // bindings
} // ipv4

container ipv6 {
    when "../..afi = 'ipv6'" {
        description
        "Only for IPv6.";
    }
    description
    "IPv6 state information.";
    container roots {
        description
        "IPv6 multicast LSP roots.";
        list root {
            key "root-address";
            description
            "List of roots for configured multicast LSPs.";

            leaf root-address {
                type inet:ipv6-address;
                description
                "Root address.";
            }

            leaf is-self {
                type boolean;
                description
                "This is the root.";
            }
        }

        list reachability {
            key "address interface";
            description
            "A next hop for reachability to root,
            as a RIB view.";
            leaf address {
                type inet:ipv6-address;
                description
                "The next hop address to reach root.";
            }
        }
    }
}
```

```
    leaf interface {
      type mpls-interface-ref;
      description
        "Interface connecting to next-hop.";
    }
    leaf peer {
      type leafref {
        path
          "../.../.../.../.../.../.../peers/peer/"
          + "lsr-id";
      }
      description
        "LDP peer from which this next hop can be
        reached.";
    }
  }
} // list root
} // roots
container bindings {
  description
    "mLDP FEC to label bindings.";
  container opaque-type-lspid {
    description
      "The type of opaque value element is
      the generic LSP identifier";
    reference
      "RFC6388: Label Distribution Protocol
      Extensions for Point-to-Multipoint and
      Multipoint-to-Multipoint Label Switched
      Paths.";
    list fec-label {
      key
        "root-address lsp-id "
        + "recur-root-address recur-rd";
      description
        "List of FEC to label bindings.";
      leaf root-address {
        type inet:ipv6-address;
        description
          "Root address.";
      }
      leaf lsp-id {
        type uint32;
        description "ID to identify the LSP.";
      }
      leaf recur-root-address {
        type inet:ip-address;
        description

```

```
        "Recursive root address.";
        reference
        "RFC6512: Using Multipoint LDP When the
        Backbone Has No Route to the Root";
    }
    leaf recur-rd {
        type route-distinguisher;
        description
        "Route Distinguisher in the VPN-Recursive
        Opaque Value.";
        reference
        "RFC6512: Using Multipoint LDP When the
        Backbone Has No Route to the Root";
    }
    uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-lspid

container opaque-type-src {
    description
    "The type of opaque value element is
    the transit Source TLV";
    reference
    "RFC6826: Multipoint LDP In-Band Signaling for
    Point-to-Multipoint and
    Multipoint-to-Multipoint Label Switched
    Paths.";
    list fec-label {
        key
        "root-address source-address group-address "
        + "rd recur-root-address recur-rd";
        description
        "List of FEC to label bindings.";
        leaf root-address {
            type inet:ipv6-address;
            description
            "Root address.";
        }
        leaf source-address {
            type inet:ip-address;
            description
            "Source address.";
        }
        leaf group-address {
            type inet:ip-address-no-zone;
            description
            "Group address.";
        }
    }
}
```

```
leaf rd {
  type route-distinguisher;
  description
    "Route Distinguisher.";
  reference
    "RFC7246: Multipoint Label Distribution
    Protocol In-Band Signaling in a Virtual
    Routing and Forwarding (VRF) Table
    Context.";
}
leaf recur-root-address {
  type inet:ip-address;
  description
    "Recursive root address.";
  reference
    "RFC6512: Using Multipoint LDP When the
    Backbone Has No Route to the Root";
}
leaf recur-rd {
  type route-distinguisher;
  description
    "Route Distinguisher in the VPN-Recursive
    Opaque Value.";
  reference
    "RFC6512: Using Multipoint LDP When the
    Backbone Has No Route to the Root";
}
uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-src

container opaque-type-bidir {
  description
    "The type of opaque value element is
    the generic LSP identifier";
  reference
    "RFC6826: Multipoint LDP In-Band Signaling for
    Point-to-Multipoint and
    Multipoint-to-Multipoint Label Switched
    Paths.";
  list fec-label {
    key
      "root-address rp group-address "
      + "rd recur-root-address recur-rd";
    description
      "List of FEC to label bindings.";
    leaf root-address {
      type inet:ipv6-address;
```

```
        description
            "Root address.";
    }
    leaf rp {
        type inet:ip-address;
        description
            "RP address.";
    }
    leaf group-address {
        type inet:ip-address-no-zone;
        description
            "Group address.";
    }
    leaf rd {
        type route-distinguisher;
        description
            "Route Distinguisher.";
        reference
            "RFC7246: Multipoint Label Distribution
            Protocol In-Band Signaling in a Virtual
            Routing and Forwarding (VRF) Table
            Context.";
    }
    leaf recur-root-address {
        type inet:ip-address;
        description
            "Recursive root address.";
        reference
            "RFC6512: Using Multipoint LDP When the
            Backbone Has No Route to the Root";
    }
    leaf recur-rd {
        type route-distinguisher;
        description
            "Route Distinguisher in the VPN-Recursive
            Opaque Value.";
        reference
            "RFC6512: Using Multipoint LDP When the
            Backbone Has No Route to the Root";
    }
    uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-bidir
} // bindings
} // ipv6
} // state

container configured-leaf-lsps {
```

```
    description
      "Configured multicast LSPs.";

    container p2mp {
      description
        "Configured point-to-multipoint LSPs.";
      uses mldp-configured-lsp-roots;
    }
    container mp2mp {
      description
        "Configured multipoint-to-multipoint LSPs.";
      uses mldp-configured-lsp-roots;
    }
  } // configured-leaf-lsps
} // list address-family
} // mldp

list address-family {
  key "afi";
  description
    "Per-vrf per-af params.";
  leaf afi {
    type ldp-address-family;
    description
      "Address family type value.";
  }
}

container config {
  description
    "Configuration data.";
  leaf enable {
    type boolean;
    description
      "'true' to enable the address family.";
  }
  uses policy-container;

  container ipv4 {
    when "../..//afi = 'ipv4'" {
      description
        "Only for IPv4.";
    }
    description
      "IPv4 address family.";
    leaf transport-address {
      type inet:ipv4-address;
      description
        "The transport address advertised in LDP Hello
```

```
        messages.";
    }
} // ipv4
container ipv6 {
    when "../..afi = 'ipv6'" {
        description
            "Only for IPv6.";
    }
    description
        "IPv6 address family.";
    leaf transport-address {
        type inet:ipv6-address;
        description
            "The transport address advertised in LDP Hello
            messages.";
    }
} // ipv6
}
container state {
    config false;
    description
        "Operational state data.";
    leaf enable {
        type boolean;
        description
            "'true' to enable the address family.";
    }
}

uses policy-container;

container ipv4 {
    when "../..afi = 'ipv4'" {
        description
            "Only for IPv4.";
    }
    description
        "IPv4 address family.";
    leaf transport-address {
        type inet:ipv4-address;
        description
            "The transport address advertised in LDP Hello
            messages.";
    }
}

container bindings {
    description
        "LDP address and label binding information.";
    list address {
```



```
    key "address";
    description
        "List of address bindings.";
    leaf address {
        type inet:ipv4-address;
        description
            "Binding address.";
    }
    uses binding-address-state-attributes;
} // binding-address

list fec-label {
    key "fec";
    description
        "List of label bindings.";
    leaf fec {
        type inet:ipv4-prefix;
        description
            "Prefix FEC.";
    }
    uses binding-label-state-attributes;
} // fec-label
} // binding
} // ipv4
container ipv6 {
    when "../..afi = 'ipv6'" {
        description
            "Only for IPv6.";
    }
    description
        "IPv6 address family.";
    leaf transport-address {
        type inet:ipv6-address;
        description
            "The transport address advertised in LDP Hello
            messages.";
    }
}

container binding {
    description
        "LDP address and label binding information.";
    list address {
        key "address";
        description
            "List of address bindings.";
        leaf address {
            type inet:ipv6-address;
            description
```

```
        "Binding address.";
    }
    uses binding-address-state-attributes;
} // binding-address

list fec-label {
    key "fec";
    description
        "List of label bindings.";
    leaf fec {
        type inet:ipv6-prefix;
        description
            "Prefix FEC.";
    }
    uses binding-label-state-attributes;
} // fec-label
} // binding
} // ipv6
} // state
} // address-family

container discovery {
    description
        "Neighbor discovery configuration.";

    container interfaces {
        description
            "A list of interfaces for basic discovery.";
        container config {
            description
                "Configuration data.";
            uses basic-discovery-timers;
        }
        container state {
            config false;
            description

                "Operational state data.";
            uses basic-discovery-timers;
        }
    }

    list interface {
        key "interface";
        description
            "List of LDP interfaces.";
        leaf interface {
            type mpls-interface-ref;
            description
```

```
        "Interface.";
    }
    container config {
        description
            "Configuration data.";
        uses basic-discovery-timers {
            if-feature per-interface-timer-config;
        }
        leaf igp-synchronization-delay {
            if-feature per-interface-timer-config;
            type uint16 {
                range 3..60;
            }
            units seconds;
            description
                "Sets the interval that the LDP waits before
                 notifying the Interior Gateway Protocol (IGP)
                 that label exchange is completed so that IGP
                 can start advertising the normal metric for
                 the link.";
        }
    }
    container state {
        config false;
        description
            "Operational state data.";
        uses basic-discovery-timers {
            if-feature per-interface-timer-config;
        }
        leaf igp-synchronization-delay {
            if-feature per-interface-timer-config;
            type uint16 {
                range 3..60;
            }
            units seconds;
            description
                "Sets the interval that the LDP waits before
                 notifying the Interior Gateway Protocol (IGP)
                 that label exchange is completed so that IGP
                 can start advertising the normal metric for
                 the link.";
        }
        leaf next-hello {
            type uint16;
            units seconds;
            description "Time to send the next hello message.";
        }
    }
} // state
```

```
list address-family {
  key "afi";
  description
    "Per-vrf per-af params.";
  leaf afi {
    type ldp-address-family;
    description
      "Address family type value.";
  }
  container config {
    description
      "Configuration data.";
    leaf enable {
      type boolean;
      description
        "Enable the address family on the interface.";
    }
  }

  container ipv4 {
    must "/if:interfaces/if:interface"
      + "[name = current()/../../../../interface]/"
      + "ip:ipv4" {
      description
        "Only if IPv4 is enabled on the interface.";
    }
    description
      "IPv4 address family.";
    leaf transport-address {
      type union {
        type enumeration {
          enum "use-interface-address" {
            description
              "Use interface address as the transport
              address.";
          }
        }
        type inet:ipv4-address;
      }
    }
    description
      "IP address to be advertised as the LDP
      transport address.";
  }
}

container ipv6 {
  must "/if:interfaces/if:interface"
    + "[name = current()/../../../../interface]/"
    + "ip:ipv6" {
  }
```

```
        description
            "Only if IPv6 is enabled on the interface.";
    }
    description
        "IPv6 address family.";
    leaf transport-address {
        type union {
            type enumeration {
                enum "use-interface-address" {
                    description
                        "Use interface address as the transport
                        address.";
                }
            }
            type inet:ipv4-address;
        }
        description
            "IP address to be advertised as the LDP
            transport address.";
    }
} // ipv6
}
container state {
    config false;
    description
        "Operational state data.";
    leaf enable {
        type boolean;
        description
            "Enable the address family on the interface.";
    }
}

container ipv4 {
    must "/if:interfaces/if:interface"
        + "[name = current()/../../../interface]/"
        + "ip:ipv4" {
        description
            "Only if IPv4 is enabled on the interface.";
    }
    description
        "IPv4 address family.";
    leaf transport-address {
        type union {
            type enumeration {

                enum "use-interface-address" {
                    description
                        "Use interface address as the transport
```

```
        address.";
    }
}
type inet:ipv4-address;
}
description
    "IP address to be advertised as the LDP
    transport address.";
}

list hello-adjacencies {
    key "adjacent-address";
    description "List of hello adjacencies.";

    leaf adjacent-address {
        type inet:ipv4-address;
        description
            "Neighbor address of the hello adjacency.";
    }

    uses adjacency-state-attributes;

    leaf peer {
        type leafref {
            path "../.../.../.../.../.../peers/peer/"
                + "lsr-id";
        }
        description
            "LDP peer from this adjacency.";
    }
} // hello-adjacencies
}

container ipv6 {
    must "/if:interfaces/if:interface"
        + "[name = current()/../.../.../interface]/"
        + "ip:ipv6" {
        description
            "Only if IPv6 is enabled on the interface.";
    }
}
description
    "IPv6 address family.";
leaf transport-address {
    type union {
        type enumeration {
            enum "use-interface-address" {
                description
                    "Use interface address as the transport
                    address.";
            }
        }
    }
}
```

```

    }
  }
  type inet:ipv4-address;
}
description
  "IP address to be advertised as the LDP
  transport address.";
}

list hello-adjacencies {
  key "adjacent-address";
  description "List of hello adjacencies.";

  leaf adjacent-address {
    type inet:ipv6-address;
    description
      "Neighbor address of the hello adjacency.";
  }

  uses adjacency-state-attributes;

  leaf peer {
    type leafref {
      path "../../../../../peers/peer/"
        + "lsr-id";
    }
    description
      "LDP peer from this adjacency.";
  }
} // hello-adjacencies
} // ipv6
}
} // address-family
} // list interface
} // interfaces

container targeted
{
  description
    "A list of targeted neighbors for extended discovery.";
  container config {

    description
      "Configuration data.";
    uses extended-discovery-timers;
    uses extended-discovery-policy-attributes;
  }
  container state {

```

```
    config false;
    description
      "Operational state data.";
    uses extended-discovery-timers;
    uses extended-discovery-policy-attributes;
  }

  list address-family {
    key "afi";
    description
      "Per-af params.";
    leaf afi {
      type ldp-address-family;
      description
        "Address family type value.";
    }
  }

  container state {
    config false;
    description
      "Operational state data.";

    container ipv4 {
      when "../../afi = 'ipv4'" {
        description
          "For IPv4.";
      }
      description
        "IPv4 address family.";
      list hello-adjacencies {
        key "local-address adjacent-address";
        description "List of hello adjacencies.";

        leaf local-address {
          type inet:ipv4-address;
          description
            "Local address of the hello adjacency.";
        }
        leaf adjacent-address {
          type inet:ipv4-address;
          description
            "Neighbor address of the hello adjacency.";
        }
      }

      uses adjacency-state-attributes;

      leaf peer {
        type leafref {
```



```

        path "../../../peers/peer/"
          + "lsr-id";
      }
      description
        "LDP peer from this adjacency.";
    }
  } // hello-adjacencies
} // ipv4

container ipv6 {
  when "../../../afi = 'ipv6'" {
    description
      "For IPv6.";
  }
  description
    "IPv6 address family.";
  list hello-adjacencies {
    key "local-address adjacent-address";
    description "List of hello adjacencies.";

    leaf local-address {
      type inet:ipv6-address;
      description
        "Local address of the hello adjacency.";
    }
    leaf adjacent-address {
      type inet:ipv6-address;
      description
        "Neighbor address of the hello adjacency.";
    }
  }

  uses adjacency-state-attributes;

  leaf peer {
    type leafref {
      path "../../../peers/peer/"
        + "lsr-id";
    }
    description
      "LDP peer from this adjacency.";
  }
} // hello-adjacencies
} // ipv6
} // state

container ipv4 {
  when "../../../afi = 'ipv4'" {
    description

```

```
        "For IPv4.";
    }
    description
        "IPv4 address family.";
    list target {
        key "adjacent-address";
        description
            "Targeted discovery params.";

        leaf adjacent-address {
            type inet:ipv4-address;
            description
                "Configures a remote LDP neighbor and enables
                 extended LDP discovery of the specified
                 neighbor.";
        }
        container config {
            description
                "Configuration data.";
            leaf enable {
                type boolean;
                description
                    "Enable the target.";
            }
            leaf local-address {
                type inet:ipv4-address;
                description
                    "The local address.";
            }
        }
    }
    container state {
        config false;
        description
            "Operational state data.";
        leaf enable {
            type boolean;
            description
                "Enable the target.";
        }
        leaf local-address {
            type inet:ipv4-address;
            description
                "The local address.";
        }
    }
    } // state
} // ipv4
container ipv6 {
```

```
when "../afi = 'ipv6'" {
  description
    "For IPv6.";
}
description
  "IPv6 address family.";
list target {
  key "adjacent-address";
  description
    "Targeted discovery params.";

  leaf adjacent-address {
    type inet:ipv6-address;
    description
      "Configures a remote LDP neighbor and enables
       extended LDP discovery of the specified
       neighbor.";
  }
  container config {
    description
      "Configuration data.";
    leaf enable {
      type boolean;
      description
        "Enable the target.";
    }
    leaf local-address {
      type inet:ipv6-address;
      description
        "The local address.";
    }
  }
}
container state {
  config false;
  description
    "Operational state data.";
  leaf enable {
    type boolean;
    description
      "Enable the target.";
  }
  leaf local-address {
    type inet:ipv6-address;
    description
      "The local address.";
  }
} // state
}
```

```
        } // ipv6
      } // address-family
    } // targeted
  } // discovery

  container forwarding-nexthop {
    if-feature forwarding-nexthop-config;
    description
      "Configuration for forwarding nexthop.";

    container interfaces {
      description
        "A list of interfaces on which forwarding is
        disabled.";

      list interface {
        key "interface";
        description
          "List of LDP interfaces.";
        leaf interface {
          type mpls-interface-ref;
          description
            "Interface.";
        }
      }
      list address-family {
        key "afi";
        description
          "Per-vrf per-af params.";
        leaf afi {
          type ldp-address-family;
          description
            "Address family type value.";
        }
      }
      container config {
        description
          "Configuration data.";
        leaf ldp-disable {
          type boolean;
          description
            "Disable LDP forwarding on the interface.";
        }
        leaf mldp-disable {
          if-feature mldp;
          type boolean;
          description
            "Disable mLDP forwarding on the interface.";
        }
      }
    }
  }
```

```
        container state {
            config false;
            description
                "Operational state data.";
            leaf ldp-disable {
                type boolean;
                description
                    "Disable LDP forwarding on the interface.";
            }
            leaf mldp-disable {
                if-feature mldp;

                type boolean;
                description
                    "Disable mLDP forwarding on the interface.";
            }
        }
    } // address-family
} // list interface
} // interfaces
} // forwarding-nexthop
uses policy-container {
    if-feature all-af-policy-config;
}
} // global

container peers {
    description
        "Peers configuration attributes.";

    container config {
        description
            "Configuration data.";
        uses peer-authentication {
            if-feature global-session-authentication;
        }
        uses peer-attributes;

        container session-downstream-on-demand {
            if-feature session-downstream-on-demand-config;
            description
                "Session downstream-on-demand attributes.";
            leaf enable {
                type boolean;
                description
                    "'true' if session downstream-on-demand is enabled.";
            }
            leaf peer-list {
```

```
        type peer-list-ref;
        description
            "The name of a peer ACL.";
    }
}
}
container state {
    config false;
    description
        "Operational state data.";
    uses peer-authentication {
        if-feature global-session-authentication;
    }
    uses peer-attributes;

    container session-downstream-on-demand {
        if-feature session-downstream-on-demand-config;
        description
            "Session downstream-on-demand attributes.";
        leaf enable {
            type boolean;
            description
                "'true' if session downstream-on-demand is enabled.";
        }
        leaf peer-list {
            type peer-list-ref;
            description
                "The name of a peer ACL.";
        }
    }
}

list peer {
    key "lsr-id";
    description
        "List of peers.";

    leaf lsr-id {
        type yang:dotted-quad;
        description "LSR ID.";
    }

    container config {
        description
            "Configuration data.";
        leaf admin-down {
            type boolean;
            default false;
        }
    }
}
```

```
        description
            "'true' to disable the peer.";
    }

    container capability {
        description
            "Per peer capability";
        container mldp {
            if-feature mldp;
            description
                "mLDP capabilities.";
            uses mldp-capabilities;
        }
    }

    uses peer-af-policy-container {
        if-feature all-af-policy-config;
    }

    uses peer-authentication;

    uses graceful-restart-attributes-per-peer {
        if-feature per-peer-graceful-restart-config;
    }

    uses peer-attributes {
        if-feature per-peer-session-attributes-config;
    }

    container address-family {
        description
            "Per-vrf per-af params.";
        container ipv4 {
            description
                "IPv4 address family.";
            uses peer-af-policy-container;
        }
        container ipv6 {
            description
                "IPv6 address family.";
            uses peer-af-policy-container;
        } // ipv6
    } // address-family
}

container state {
    config false;
    description
        "Operational state data.";
```

```
leaf admin-down {
    type boolean;
    default false;
    description
        "'true' to disable the peer.";
}

container capability {
    description
        "Per peer capability";
    container mldp {
        if-feature mldp;
        description
            "mLDP capabilities.";
        uses mldp-capabilities;
    }
}

uses peer-af-policy-container {
    if-feature all-af-policy-config;
}

uses peer-authentication;

uses graceful-restart-attributes-per-peer {
    if-feature per-peer-graceful-restart-config;
}

uses peer-attributes {
    if-feature per-peer-session-attributes-config;
}

container address-family {
    description
        "Per-vrf per-af params.";
    container ipv4 {
        description
            "IPv4 address family.";
        uses peer-af-policy-container;

        list hello-adjacencies {
            key "local-address adjacent-address";
            description "List of hello adjacencies.";

            leaf local-address {
                type inet:ipv4-address;
                description
                    "Local address of the hello adjacency.";
            }
        }
    }
}
```



```
    }
    leaf adjacent-address {
      type inet:ipv4-address;
      description
        "Neighbor address of the hello adjacency.";
    }

    uses adjacency-state-attributes;

    leaf interface {
      type mpls-interface-ref;
      description "Interface for this adjacency.";
    }
  } // hello-adjacencies
} // ipv4
container ipv6 {
  description
    "IPv6 address family.";
  uses peer-af-policy-container;

  list hello-adjacencies {
    key "local-address adjacent-address";
    description "List of hello adjacencies.";

    leaf local-address {
      type inet:ipv6-address;
      description
        "Local address of the hello adjacency.";
    }
    leaf adjacent-address {
      type inet:ipv6-address;
      description
        "Neighbor address of the hello adjacency.";
    }
  }

  uses adjacency-state-attributes;

  leaf interface {
    type mpls-interface-ref;
    description "Interface for this adjacency.";
  }
} // hello-adjacencies
} // ipv6
} // address-family

uses peer-state-derived;
} // state
} // list peer
```

```
    } // peers
  } // container mpls-ldp
}

/*
 * RPCs
 */
rpc mpls-ldp-clear-peer {
  description
    "Clears the session to the peer.";
  input {
    leaf lsr-id {
      type union {
        type yang:dotted-quad;
        type uint32;
      }
      description
        "LSR ID of peer to be cleared. If this is not provided
        then all peers are cleared";
    }
  }
}

rpc mpls-ldp-clear-hello-adjacency {
  description
    "Clears the hello adjacency";
  input {
    container hello-adjacency {
      description
        "Link adjacency or targettted adjacency. If this is not
        provided then all hello adjacencies are cleared";
      choice hello-adjacency-type {
        description "Adjacency type.";
        case targeted {
          container targeted {
            presence "Present to clear targeted adjacencies.";
            description
              "Clear targeted adjacencies.";
            leaf target-address {
              type inet:ip-address;
              description
                "The target address. If this is not provided then
                all targeted adjacencies are cleared";
            }
          } // targeted
        }
        case link {
          container link {

```

```

        presence "Present to clear link adjacencies.";
        description
            "Clear link adjacencies.";
        leaf next-hop-interface {
            type mpls-interface-ref;
            description
                "Interface connecting to next-hop. If this is not
                provided then all link adjacencies are cleared.";
        }
        leaf next-hop-address {
            type inet:ip-address;
            must "../next-hop-interface" {
                description
                    "Applicable when interface is specified.";
            }
            description
                "IP address of next-hop. If this is not provided
                then adjacencies to all next-hops on the given
                interface are cleared.";
        } // next-hop-address
    } // link
}
}
}
}
}

rpc mpls-ldp-clear-peer-statistics {
    description
        "Clears protocol statistics (e.g. sent and received
        counters).";
    input {
        leaf lsr-id {
            type union {
                type yang:dotted-quad;
                type uint32;
            }
            description
                "LSR ID of peer whose statistic are to be cleared.
                If this is not provided then all peers statistics are
                cleared";
        }
    }
}

/*
 * Notifications

```

```
*/
notification mpls-ldp-peer-event {
    description
        "Notification event for a change of LDP peer operational
        status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-peer-ref;
}

notification mpls-ldp-hello-adjacency-event {
    description
        "Notification event for a change of LDP adjacency operational
        status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-adjacency-ref;
}

notification mpls-ldp-fec-event {
    description
        "Notification event for a change of FEC status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-fec-event;
}

notification mpls-mldp-fec-event {
    description
        "Notification event for a change of FEC status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses mldp-fec-event;
}
}

<CODE ENDS>
```

Figure 22

## 7. Security Considerations

The configuration, state, action and notification data defined using YANG data models in this document are likely to be accessed via the protocols such as NETCONF [RFC6241] etc.

Hence, YANG implementations MUST comply with the security requirements specified in section 15 of [RFC6020]. Additionally, NETCONF implementations MUST comply with the security requirements specified in sections 2.2, 2.3 and 9 of [RFC6241] as well as section 3.7 of [RFC6536].

## 8. IANA Considerations

This document does not extend LDP or mLDP base protocol specification and hence there are no IANA considerations.

Note to the RFC Editor: Please remove IANA section before the publication.

## 9. Acknowledgments

The authors would like to acknowledge Eddie Chami, Nagendra Kumar, Mannan Venkatesan, Pavan Beeram for their contribution to this document. We also acknowledge Ladislav Lhotka for his useful comments as the YANG Doctor.

## 10. References

### 10.1. Normative References

- [I-D.ietf-netmod-routing-cfg]  
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-22 (work in progress), July 2016.
- [I-D.rtgyangdt-rtgwg-ni-model]  
Berger, L., Hopps, C., Lindem, A., and D. Bogdanovic, "Network Instance Model", draft-rtgyangdt-rtgwg-ni-model-00 (work in progress), May 2016.
- [I-D.saad-mpls-base-yang]  
Raza, K., Gandhi, R., Liu, X., Beeram, V., Saad, T., Bryskin, I., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for MPLS Base", draft-saad-mpls-base-yang-00 (work in progress), May 2016.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3478] Leelanivas, M., Rekhter, Y., and R. Aggarwal, "Graceful Restart Mechanism for Label Distribution Protocol", RFC 3478, DOI 10.17487/RFC3478, February 2003, <<http://www.rfc-editor.org/info/rfc3478>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<http://www.rfc-editor.org/info/rfc5036>>.
- [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <<http://www.rfc-editor.org/info/rfc5331>>.
- [RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, DOI 10.17487/RFC5561, July 2009, <<http://www.rfc-editor.org/info/rfc5561>>.
- [RFC5918] Asati, R., Minei, I., and B. Thomas, "Label Distribution Protocol (LDP) 'Typed Wildcard' Forward Equivalence Class (FEC)", RFC 5918, DOI 10.17487/RFC5918, August 2010, <<http://www.rfc-editor.org/info/rfc5918>>.
- [RFC5919] Asati, R., Mohapatra, P., Chen, E., and B. Thomas, "Signaling LDP Label Advertisement Completion", RFC 5919, DOI 10.17487/RFC5919, August 2010, <<http://www.rfc-editor.org/info/rfc5919>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<http://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<http://www.rfc-editor.org/info/rfc6241>>.

- [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <<http://www.rfc-editor.org/info/rfc6388>>.
- [RFC6389] Aggarwal, R. and JL. Le Roux, "MPLS Upstream Label Assignment for LDP", RFC 6389, DOI 10.17487/RFC6389, November 2011, <<http://www.rfc-editor.org/info/rfc6389>>.
- [RFC6512] Wijnands, IJ., Rosen, E., Napierala, M., and N. Leymann, "Using Multipoint LDP When the Backbone Has No Route to the Root", RFC 6512, DOI 10.17487/RFC6512, February 2012, <<http://www.rfc-editor.org/info/rfc6512>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<http://www.rfc-editor.org/info/rfc6536>>.
- [RFC6826] Wijnands, IJ., Ed., Eckert, T., Leymann, N., and M. Napierala, "Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6826, DOI 10.17487/RFC6826, January 2013, <<http://www.rfc-editor.org/info/rfc6826>>.
- [RFC7060] Napierala, M., Rosen, E., and IJ. Wijnands, "Using LDP Multipoint Extensions on Targeted LDP Sessions", RFC 7060, DOI 10.17487/RFC7060, November 2013, <<http://www.rfc-editor.org/info/rfc7060>>.
- [RFC7140] Jin, L., Jounay, F., Wijnands, IJ., and N. Leymann, "LDP Extensions for Hub and Spoke Multipoint Label Switched Path", RFC 7140, DOI 10.17487/RFC7140, March 2014, <<http://www.rfc-editor.org/info/rfc7140>>.
- [RFC7246] Wijnands, IJ., Ed., Hitchen, P., Leymann, N., Henderickx, W., Gulko, A., and J. Tantsura, "Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context", RFC 7246, DOI 10.17487/RFC7246, June 2014, <<http://www.rfc-editor.org/info/rfc7246>>.
- [RFC7438] Wijnands, IJ., Ed., Rosen, E., Gulko, A., Joerde, U., and J. Tantsura, "Multipoint LDP (mLDP) In-Band Signaling with Wildcards", RFC 7438, DOI 10.17487/RFC7438, January 2015, <<http://www.rfc-editor.org/info/rfc7438>>.

- [RFC7552] Asati, R., Pignataro, C., Raza, K., Manral, V., and R. Papneja, "Updates to LDP for IPv6", RFC 7552, DOI 10.17487/RFC7552, June 2015, <<http://www.rfc-editor.org/info/rfc7552>>.
- [RFC7715] Wijnands, IJ., Ed., Raza, K., Atlas, A., Tantsura, J., and Q. Zhao, "Multipoint LDP (mLDP) Node Protection", RFC 7715, DOI 10.17487/RFC7715, January 2016, <<http://www.rfc-editor.org/info/rfc7715>>.

## 10.2. Informative References

- [I-D.ietf-rtgwg-policy-model]  
Shaikh, A., Shakir, R., D'Souza, K., and C. Chase,  
"Routing Policy Configuration Model for Service Provider  
Networks", draft-ietf-rtgwg-policy-model-01 (work in  
progress), April 2016.
- [I-D.iwijnand-mpls-mlbp-multi-topology]  
Wijnands, I. and K. Raza, "mLDP Extensions for Multi  
Topology Routing", draft-iwijnand-mpls-mlbp-multi-  
topology-03 (work in progress), June 2013.
- [I-D.openconfig-netmod-opstate]  
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling  
of Operational State Data in YANG", draft-openconfig-  
netmod-opstate-01 (work in progress), July 2015.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private  
Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February  
2006, <<http://www.rfc-editor.org/info/rfc4364>>.
- [RFC7307] Zhao, Q., Raza, K., Zhou, C., Fang, L., Li, L., and D.  
King, "LDP Extensions for Multi-Topology", RFC 7307,  
DOI 10.17487/RFC7307, July 2014,  
<<http://www.rfc-editor.org/info/rfc7307>>.

## Appendix A. Additional Contributors

Stephane Litkowski  
Orange.  
Email: [stephane.litkowski@orange.com](mailto:stephane.litkowski@orange.com)

Reshad Rahman  
Cisco Systems Inc.  
Email: [rrahman@cisco.com](mailto:rrahman@cisco.com)

Danial Johari  
Cisco Systems Inc.  
Email: [dajohari@cisco.com](mailto:dajohari@cisco.com)

## Authors' Addresses

Kamran Raza  
Cisco Systems, Inc.  
Email: [skraza@cisco.com](mailto:skraza@cisco.com)

Rajiv Asati  
Cisco Systems, Inc.  
Email: [rajiva@cisco.com](mailto:rajiva@cisco.com)



Sowmya Krishnaswamy  
Cisco Systems, Inc.  
Email: sowkrish@cisco.com

Xufeng Liu  
Ericsson  
Email: xliu@kuatrotech.com

Jeff Tantsura  
Ericsson  
Email: jeff.tantsura@ericsson.com

Santosh Esale  
Juniper Networks  
Email: sesale@juniper.net

Xia Chen  
Huawei Technologies  
Email: jescia.chenxia@huawei.com

Loa Andersson  
Huawei Technologies  
Email: loa@pi.nu

Himanshu Shah  
Ciena Corporation  
Email: hshah@ciena.com

Matthew Bocci  
Alcatel-Lucent  
Email: matthew.bocci@alcatel-lucent.com

MPLS Working Group  
Internet-Draft  
Updates: 7271 (if approved)  
Intended status: Standards Track  
Expires: September 4, 2016

J. Ryoo  
T. Cheung  
ETRI  
H. van Helvoort  
Hai Gaoming BV  
I. Busi  
G. Weng  
Huawei Technologies  
March 3, 2016

Updates to MPLS Transport Profile (MPLS-TP) Linear Protection in  
Automatic Protection Switching (APS) Mode  
draft-ryoo-mpls-tp-aps-updates-03.txt

#### Abstract

This document contains updates to MPLS Transport Profile (MPLS-TP) linear protection in Automatic Protection Switching (APS) mode defined in RFC 7271. The updates provide rules related to the initialization of the Protection State Coordination (PSC) Control Logic, in which the state machine resides, when operating in APS mode.

#### 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 <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 4, 2016.

#### Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction . . . . .	2
2. Conventions Used in This Document . . . . .	3
3. Acronyms . . . . .	3
4. Updates . . . . .	4
4.1. Initialization Behavior . . . . .	4
4.2. State Transition Modification . . . . .	5
5. Security Considerations . . . . .	5
6. IANA Considerations . . . . .	6
7. Acknowledgements . . . . .	6
8. References . . . . .	6
8.1. Normative References . . . . .	6
8.2. Informative References . . . . .	6
Authors' Addresses . . . . .	6

## 1. Introduction

MPLS Transport Profile (MPLS-TP) linear protection in Automatic Protection Switching (APS) mode is defined in RFC 7271 [RFC7271]. It defines a set of alternate and additional mechanisms to perform some of the functions of linear protection described in RFC 6378 [RFC6378]. The actions performed at initialization of the Protection State Coordination (PSC) Control Logic are not described in either [RFC7271] or [RFC6378]. Although it is a common perception that the state machine starts at the Normal state, this is not explicitly specified in any of the documents and various questions have been raised by implementers and in discussions on the MPLS working group mailing list concerning the detailed actions that the PSC Control Logic should take.

The state machine described in [RFC7271] operates under the assumption that both end nodes of a linear protection domain start in the Normal state. In the case that one node reboots while the other node is still in operation, various scenarios may arise resulting in problematic situations. This document resolves all the problematic cases and minimizes traffic disruptions related to initialization including both cold and warm reboots that require re-initialization of the PSC Control Logic.

This document contains updates to the MPLS-TP linear protection in APS mode defined in [RFC7271]. The updates provide rules related to initialization of the PSC Control Logic, in which the state machine resides, when operating in APS mode. The updates also include modifications to the state transition table defined in Section 11.2 of [RFC7271]. The changes in the state transition table have been examined to make sure that they do not introduce any new problems.

This document does not introduce backward compatibility issues with implementations of [RFC7271]. In case a node implementing this document restarts, the new state changes will not cause problems at the remote node implementing [RFC7271] and the two ends will converge to the same local and remote states. In case a node implementing [RFC7271] restarts, the two ends behave as today.

The reader of this document is assumed to be familiar with [RFC7271].

## 2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

## 3. Acronyms

This document uses the following acronyms:

APS	Automatic Protection Switching
DNR	Do-not-Revert
E::R	Exercise state due to remote EXER message
EXER	Exercise
MS-P	Manual Switch to Protection path
MS-W	Manual Switch to Working path
MPLS-TP	MPLS Transport Profile
N	Normal state
NR	No Request
PF:DW:R	Protecting Failure state due to remote SD-W message
PF:W:L	Protecting Failure state due to local SF-W
PF:W:R	Protecting Failure state due to remote SF-W message
PSC	Protection State Coordination
RR	Reverse Request
SD	Signal Degrade
SF-P	Signal Fail on Protection path
SF-W	Signal Fail on Working path
UA:P:L	Unavailable state due to local SF-P
WTR	Wait-to-Restore

#### 4. Updates

This document updates [RFC7271] by specifying the actions that will be performed at the initialization of the PSC Control Logic and modifies the state transition table defined in Section 11.2 of [RFC7271].

##### 4.1. Initialization Behavior

This section defines initialization behavior that is not described in [RFC7271].

When the PSC Control Logic is initialized, the following actions MUST be performed:

- o Stop the WTR timer if it is running.
- o Clear any operator command in the Local Request Logic.
- o If an SF-W or SF-P exists as the highest local request, the node being initialized starts at the PF:W:L or UA:P:L state, respectively.
- o If the node being initialized has no local request:
  - \* If the node being initialized does not remember the active path or if the node being initialized remembers the working path as the active path, the node starts at the Normal state.
  - \* Else (the node being initialized remembers the protection path as the active path), the node starts at the WTR state sending NR(0,1) or at the DNR state sending DNR(0,1) depending on the configuration that allows or prevents automatic reversion to the Normal state.
- o In case any local SD exists, the local SD MUST be considered as an input to the Local Request Logic only after the local node has received the first protocol message from the remote node and completed the processing (i.e., updated the PSC Control Logic and decided which action, if any, to be sent to the PSC Message Generator).
- o If the local node receives an EXER message as the first protocol message after initialization and the remote EXER becomes the top-priority global request, the local node MUST set the position of the bridge and selector according to the Path value in the EXER message and transit to the E::R state.

Remembering the active path in case of no local request minimizes traffic switchovers in cases where the remote node is still in operation. This approach does not cause a problem even if the remembered active path is no longer valid due to any local input that occurred at the remote node while the initializing node was out of operation.

It is worth noting that in some restart scenarios (e.g., cold rebooting) no valid SF/SD indications may be present at the input of the Local Request logic. In this case, the PSC Control Logic would restart as if no local requests are present. If a valid SF/SD indication is detected later, this would be notified to the PSC Control Logic and trigger state change.

#### 4.2. State Transition Modification

In addition to the initialization behavior described in Section 4.1, four cells of the remote state transition table need to be changed to make two end nodes converge after initialization. State transition by remote message defined in Section 11.2 of [RFC7271] is modified as follows (only modified cells are shown):

	MS-W	MS-P	WTR	EXER	RR	DNR	NR
N			(13)			DNR	
PF:W:R						DNR	
PF:DW:R						DNR	

The changes in two rows of remote protecting failure states lead to the replacement of note (10) with DNR, therefore note (10) is no longer needed. The resultant three rows read:

	MS-W	MS-P	WTR	EXER	RR	DNR	NR
N	SA:MW:R	SA:MP:R	(13)	E::R	i	DNR	i
PF:W:R	SA:MW:R	SA:MP:R	(9)	E::R	i	DNR	(11)
PF:DW:R	SA:MW:R	SA:MP:R	(9)	E::R	i	DNR	(11)

#### 5. Security Considerations

No specific security issue is raised in addition to those ones already documented in [RFC7271]. It may be noted that tightening the description of initializing behavior may help to protect networks from re-start attacks.

## 6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

## 7. Acknowledgements

The authors would like to thank Joaquim Serra for bringing up the issue related to initialization of the PSC Control Logic at the very beginning. The authors would also like to thank Adrian Farrel for his valuable comments and suggestions on this document.

## 8. References

### 8.1. 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, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC7271] Ryoo, J., Ed., Gray, E., Ed., van Helvoort, H., D'Alessandro, A., Cheung, T., and E. Osborne, "MPLS Transport Profile (MPLS-TP) Linear Protection to Match the Operational Expectations of Synchronous Digital Hierarchy, Optical Transport Network, and Ethernet Transport Network Operators", RFC 7271, DOI 10.17487/RFC7271, June 2014, <<http://www.rfc-editor.org/info/rfc7271>>.

### 8.2. Informative References

- [RFC6378] Weingarten, Y., Ed., Bryant, S., Osborne, E., Sprecher, N., and A. Fulignoli, Ed., "MPLS Transport Profile (MPLS-TP) Linear Protection", RFC 6378, DOI 10.17487/RFC6378, October 2011, <<http://www.rfc-editor.org/info/rfc6378>>.

### Authors' Addresses

Jeong-dong Ryoo  
ETRI

EMail: [ryoo@etri.re.kr](mailto:ryoo@etri.re.kr)



Taesik Cheung  
ETRI

EMail: cts@etri.re.kr

Huub van Helvoort  
Hai Gaoming BV

EMail: huubatwork@gmail.com

Italo Busi  
Huawei Technologies

EMail: Italo.Busi@huawei.com

Guangjuan Weng  
Huawei Technologies

EMail: wenguangjuan@huawei.com

MPLS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: November 12, 2016

T. Saad  
K. Raza  
R. Gandhi  
Cisco Systems Inc  
X. Liu  
Ericsson  
V. Beeram  
Juniper Networks  
H. Shah  
Ciena  
I. Bryskin  
X. Chen  
Huawei Technologies  
R. Jones  
Brocade  
B. Wen  
Comcast  
May 11, 2016

A YANG Data Model for MPLS Static LSPs  
draft-saad-mpls-static-yang-03

Abstract

This document contains the specification for the MPLS Static Label Switched Paths (LSPs) YANG model. The model allows for the provisioning of static LSP(s) on LER(s) and LSR(s) devices along a LSP path without the dependency on any signaling protocol. The MPLS Static LSP model augments the MPLS base YANG model with specific data to configure and manage MPLS Static LSP(s).

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 <http://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 November 12, 2016.

## Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
1.1. Terminology . . . . .	3
1.2. MPLS Static LSPs Model Tree Diagram . . . . .	4
1.3. MPLS Static LSP YANG Module . . . . .	5
2. IANA Considerations . . . . .	11
3. Security Considerations . . . . .	11
4. References . . . . .	11
4.1. Normative References . . . . .	11
4.2. Informative References . . . . .	12
Authors' Addresses . . . . .	12

## 1. Introduction

This document describes a YANG data model for configuring and managing the Static LSPs feature. The model allows the configuration of LER and LSR devices with the necessary MPLS cross-connects or bindings to realize an end-to-end LSP service.

A static LSP is established by manually specifying incoming and outgoing MPLS label(s) and necessary forwarding information on each of the traversed Label Edge Router (LER) and Label Switched Router (LSR) devices (ingress, transit, or egress nodes) of the forwarding path.

For example, on an ingress LER device, the model is used to associate a specific Forwarding Equivalence Class (FEC) of packets- e.g. matching a specific IP prefix in a Virtual Routing or Forwarding (VRF) instance- to an MPLS outgoing label imposition, next-hop(s) and respective outgoing interface(s) to forward the packet. On an LSR device, the model is used to create a binding that swaps the incoming label with an outgoing label and forwards the packet on one or

multiple egress path(s). On an egress LER, it is used to create a binding that decapsulates the incoming MPLS label and performs forwarding based on the inner MPLS label (if present) or IP forwarding in the packet.

The MPLS Static LSP YANG model is defined in module "ietf-mpls-static" and augments the MPLS Base YANG model defined in module "ietf-mpls" in [I-D.saad-mpls-base-yang]. The approach described in [I-D.openconfig-netmod-opstate] is adopted to represent data pertaining to configuration intended, applied state and derived state data elements. Each container in the model holds a "config" and "state" sub-container. The "config" sub-container is used to represent the intended configurable parameters, and the state sub-container is used to represent both the applied configurable parameters and any derived state, such as counters or statistical information.

### 1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

The following terms are defined in [RFC6020]:

- o augment,
- o configuration data,
- o data model,
- o data node,
- o feature,
- o mandatory node,
- o module,
- o schema tree,
- o state data,
- o RPC operation.

## 1.2. MPLS Static LSPs Model Tree Diagram

The MPLS Static LSP tree diagram is shown in Figure 1.

```

module: ietf-mpls-static
augment /rt:routing/mpls:mpls:
  +--rw static-lsps
    +--rw static-lsp* [name]
      +--rw name          string
      +--rw config
        +--rw in-segment
          +--rw (type)?
            +--:(ip-prefix)
              | +--rw ip-prefix?          inet:ip-prefix
            +--:(mpls-label)
              | +--rw incoming-label?     mpls:mpls-label
            +--rw operation?              enumeration
          +--rw (out-segment)?
            +--:(simple-path)
              | +--rw next-hop?           inet:ip-address
              | +--rw outgoing-label?     mpls:mpls-label
              | +--rw outgoing-interface? if:interface-ref
            +--:(path-list)
              +--rw paths* [path-index]
                +--rw path-index          uint16
                +--rw backup-path-index?   uint16
                +--rw next-hop?           inet:ip-address
                +--rw outgoing-labels*     mpls:mpls-label
                +--rw outgoing-interface?  if:interface-ref
                +--rw loadshare?           uint16
                +--rw role?               enumeration
      +--ro state
        +--ro in-segment
          +--ro (type)?
            +--:(ip-prefix)
              | +--ro ip-prefix?          inet:ip-prefix
            +--:(mpls-label)
              | +--ro incoming-label?     mpls:mpls-label
            +--ro operation?              enumeration
          +--ro (out-segment)?
            +--:(simple-path)
              | +--ro next-hop?           inet:ip-address
              | +--ro outgoing-label?     mpls:mpls-label
              | +--ro outgoing-interface? if:interface-ref
            +--:(path-list)
              +--ro paths* [path-index]
                +--ro path-index          uint16
                +--ro backup-path-index?   uint16

```

+-ro next-hop?	inet:ip-address
+-ro outgoing-labels*	mpls:mpls-label
+-ro outgoing-interface?	if:interface-ref
+-ro loadshare?	uint16
+-ro role?	enumeration

Figure 1: MPLS Static LSP tree diagram

### 1.3. MPLS Static LSP YANG Module

<CODE BEGINS>file "ietf-mpls-static@2016-05-11.yang"

```

module ietf-mpls-static {

    namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-static";

    prefix "mpls-static";

    import ietf-mpls {
        prefix mpls;
    }

    import ietf-routing {
        prefix "rt";
    }

    import ietf-inet-types {
        prefix inet;
    }

    import ietf-interfaces {
        prefix "if";
    }

    organization "IETF MPLS Working Group";

    contact
        "WG Web:    <http://tools.ietf.org/wg/mpls/>

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

        WG Chair:   Loa Andersson
                   <mailto:loa@pi.nu>

        WG Chair:   Ross Callon
                   <mailto:rcallon@juniper.net>

        WG Chair:   George Swallow

```

```
<mailto:swallow.ietf@gmail.com>

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

Editor:   Kamran Raza
          <mailto:skraza@cisco.com>

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

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

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

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

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

Editor:   Xia Chen
          <mailto:jescia.chenxia@huawei.com>

Editor:   Raqib Jones
          <mailto:raqib@Brocade.com>

Editor:   Bin Wen
          <mailto:Bin_Wen@cable.comcast.com>;
```

description

```
"This YANG module augments the 'ietf-routing' module with basic
configuration and operational state data for MPLS static";
```

```
revision "2016-05-11" {
  description
    "Latest revision:
     - Addressed MPLS-RT review comments";
  reference "RFC 3031: A YANG Data Model for Static MPLS LSPs";
}
```

```
grouping path-basic_config {
  description "common definitions for statics";

  leaf next-hop {
    type inet:ip-address;
```

```
    description "next hop IP address for the LSP";
  }

  leaf outgoing-label {
    type mpls:mpls-label;
    description
      "label value to push at the current hop for the
      LSP";
  }

  leaf outgoing-interface {
    type if:interface-ref;
    description
      "The outgoing interface";
  }
}

grouping path-properties_config {
  description
    "MPLS path properties";
  leaf path-index {
    type uint16;
    description
      "Path identifier";
  }

  leaf backup-path-index {
    type uint32;
    description
      "Backup path identifier";
  }

  leaf next-hop {
    type inet:ip-address;
    description
      "The address of the next-hop";
  }

  leaf-list outgoing-labels {
    type mpls:mpls-label;
    ordered-by user;
    description
      "The outgoing MPLS labels to impose";
  }

  leaf outgoing-interface {
    type if:interface-ref;
  }
}
```



```
    description
      "The outgoing interface";
  }

  leaf loadshare {
    type uint16;
    description
      "This value is used to compute a loadshare to perform un-equal
      load balancing when multiple outgoing path(s) are specified. A
      share is computed as a ratio of this number to the total under
      all configured path(s).";
  }

  leaf role {
    type enumeration {
      enum PRIMARY {
        description
          "Path as primary traffic carrying";
      }
      enum BACKUP {
        description
          "Path acts as backup";
      }
      enum PRIMARY_AND_BACKUP {
        description
          "Path acts as primary and backup simultaneously";
      }
    }
    description
      "The MPLS path role";
  }
}

grouping static-lsp_config {
  description "common definitions for static LSPs";

  container in-segment {
    description
      "MPLS incoming segment";
    choice type {
      description
        "Basic FEC choice";
      case ip-prefix {
        leaf ip-prefix {
          type inet:ip-prefix;
          description "An IP prefix";
        }
      }
    }
  }
}
```

```
    case mpls-label {
      leaf incoming-label {
        type mpls:mpls-label;
        description "label value on the incoming packet";
      }
    }
  }
}

leaf operation {
  type enumeration {
    enum impose-and-forward {
      description
        "Operation impose outgoing label(s) and forward to
        next-hop";
    }
    enum pop-and-forward {
      description
        "Operation pop incoming label and forward to next-hop";
    }
    enum pop-impose-and-forward {
      description
        "Operation pop incoming label, impose one or more
        outgoing label(s) and forward to next-hop";
    }
    enum swap-and-forward {
      description
        "Operation swap incoming label, with outgoing label and
        forward to next-hop";
    }
    enum pop-and-lookup {
      description
        "Operation pop incoming label and perform a lookup";
    }
  }
  description
    "The MPLS operation to be executed on the incoming packet";
}

choice out-segment {
  description "The MPLS out-segment type choice";
  case simple-path {
    uses path-basic_config;
  }
  case path-list {
    list paths {
      key path-index;
      description

```

```

        "The list of MPLS paths associated with the FEC";
        uses path-properties_config;
    }
}
}

grouping static-lsp {
    description "grouping for top level list of static LSPs";
    container config {
        description
            "Holds the intended configuration";
        uses static-lsp_config;
    }
    container state {
        config false;
        description
            "Holds the state and inuse configuration";
        uses static-lsp_config;
    }
}

augment "/rt:routing/mpls:mpls" {
    description "Augmentations for MPLS Static LSPs";
    container static-lsps {
        description
            "Statically configured LSPs, without dynamic signaling";
        list static-lsp {
            key name;
            description "list of defined static LSPs";

            leaf name {
                type string;
                description "name to identify the LSP";
            }
            uses static-lsp;
        }
    }
}
}

<CODE ENDS>

```

Figure 2: MPLS Static LSP YANG module

## 2. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-mpls-static XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-mpls-static namespace: urn:ietf:params:xml:ns:yang:ietf-mpls-static prefix: ietf-mpls-static reference: RFC3031

## 3. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

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

## 4. References

### 4.1. Normative References

- [I-D.saad-mpls-base-yang]  
Raza, K., Gandhi, R., Liu, X., Beeram, V., Saad, T., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for MPLS Base and Static LSPs", draft-saad-mpls-base-yang-00 (work in progress), November 2016.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.

- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<http://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<http://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<http://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, <<http://www.rfc-editor.org/info/rfc6242>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<http://www.rfc-editor.org/info/rfc6536>>.

#### 4.2. Informative References

- [I-D.openconfig-netmod-opstate]  
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-01 (work in progress), July 2015.

#### Authors' Addresses

Tarek Saad  
Cisco Systems Inc

Email: [tsaad@cisco.com](mailto:tsaad@cisco.com)

Kamran Raza  
Cisco Systems Inc

Email: [skraza@cisco.com](mailto:skraza@cisco.com)

Rakesh Gandhi  
Cisco Systems Inc

Email: [rgandhi@cisco.com](mailto:rgandhi@cisco.com)

Xufeng Liu  
Ericsson

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram  
Juniper Networks

Email: vbeeram@juniper.net

Himanshu Shah  
Ciena

Email: hshah@ciena.com

Igor Bryskin  
Huawei Technologies

Email: Igor.Bryskin@huawei.com

Xia Chen  
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones  
Brocade

Email: raqib@Brocade.com

Bin Wen  
Comcast

Email: Bin\_Wen@cable.comcast.com

Network Working Group  
Internet-Draft  
Obsoletes: 4379, 6829 (if approved)  
Intended status: Standards Track  
Expires: April 13, 2016

C. Pignataro  
N. Kumar  
Cisco  
S. Aldrin  
Google  
M. Chen  
Huawei  
October 11, 2015

Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures  
draft-smack-mpls-rfc4379bis-07

Abstract

This document describes a simple and efficient mechanism that can be used to detect data plane failures in Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs). There are two parts to this document: information carried in an MPLS "echo request" and "echo reply" for the purposes of fault detection and isolation, and mechanisms for reliably sending the echo reply.

This document obsoletes RFC 4379.

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 <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 13, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction . . . . .	3
1.1. Conventions . . . . .	4
1.2. Structure of This Document . . . . .	4
1.3. Contributors . . . . .	4
1.4. Scope of RFC4379bis work . . . . .	5
1.5. ToDo . . . . .	5
2. Motivation . . . . .	5
2.1. Use of Address Range 127/8 . . . . .	6
2.2. Router Alert Option . . . . .	8
3. Packet Format . . . . .	8
3.1. Return Codes . . . . .	13
3.2. Target FEC Stack . . . . .	13
3.2.1. LDP IPv4 Prefix . . . . .	15
3.2.2. LDP IPv6 Prefix . . . . .	15
3.2.3. RSVP IPv4 LSP . . . . .	15
3.2.4. RSVP IPv6 LSP . . . . .	16
3.2.5. VPN IPv4 Prefix . . . . .	16
3.2.6. VPN IPv6 Prefix . . . . .	17
3.2.7. L2 VPN Endpoint . . . . .	18
3.2.8. FEC 128 Pseudowire - IPv4 (Deprecated) . . . . .	18
3.2.9. FEC 128 Pseudowire - IPv4 (Current) . . . . .	19
3.2.10. FEC 129 Pseudowire - IPv4 . . . . .	20
3.2.11. BGP Labeled IPv4 Prefix . . . . .	21
3.2.12. BGP Labeled IPv6 Prefix . . . . .	21
3.2.13. Generic IPv4 Prefix . . . . .	22
3.2.14. Generic IPv6 Prefix . . . . .	22
3.2.15. Nil FEC . . . . .	23
3.2.16. FEC 128 Pseudowire - IPv6 . . . . .	23
3.2.17. FEC 129 Pseudowire - IPv6 . . . . .	24
3.3. Downstream Mapping . . . . .	25
3.3.1. Multipath Information Encoding . . . . .	28
3.3.2. Downstream Router and Interface . . . . .	30
3.4. Pad TLV . . . . .	31
3.5. Vendor Enterprise Number . . . . .	31
3.6. Interface and Label Stack . . . . .	32
3.7. Errored TLVs . . . . .	33
3.8. Reply TOS Byte TLV . . . . .	33
4. Theory of Operation . . . . .	34



4.1.	Dealing with Equal-Cost Multi-Path (ECMP)	34
4.2.	Testing LSPs That Are Used to Carry MPLS Payloads	35
4.3.	Sending an MPLS Echo Request	35
4.4.	Receiving an MPLS Echo Request	36
4.4.1.	FEC Validation	42
4.5.	Sending an MPLS Echo Reply	43
4.6.	Receiving an MPLS Echo Reply	44
4.7.	Issue with VPN IPv4 and IPv6 Prefixes	44
4.8.	Non-compliant Routers	45
5.	Security Considerations	45
6.	IANA Considerations	46
6.1.	Message Types, Reply Modes, Return Codes	47
6.2.	TLVs	47
7.	Acknowledgements	48
8.	References	49
8.1.	Normative References	49
8.2.	Informative References	50
	Authors' Addresses	51

## 1. Introduction

This document describes a simple and efficient mechanism that can be used to detect data plane failures in MPLS Label Switched Paths (LSPs). There are two parts to this document: information carried in an MPLS "echo request" and "echo reply", and mechanisms for transporting the echo reply. The first part aims at providing enough information to check correct operation of the data plane, as well as a mechanism to verify the data plane against the control plane, and thereby localize faults. The second part suggests two methods of reliable reply channels for the echo request message for more robust fault isolation.

An important consideration in this design is that MPLS echo requests follow the same data path that normal MPLS packets would traverse. MPLS echo requests are meant primarily to validate the data plane, and secondarily to verify the data plane against the control plane. Mechanisms to check the control plane are valuable, but are not covered in this document.

This document makes special use of the address range 127/8. This is an exception to the behavior defined in RFC 1122 [RFC1122] and updates that RFC. The motivation for this change and the details of this exceptional use are discussed in section 2.1 below.

### 1.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The term "Must Be Zero" (MBZ) is used in object descriptions for reserved fields. These fields MUST be set to zero when sent and ignored on receipt.

Terminology pertaining to L2 and L3 Virtual Private Networks (VPNs) is defined in [RFC4026].

Since this document refers to the MPLS Time to Live (TTL) far more frequently than the IP TTL, the authors have chosen the convention of using the unqualified "TTL" to mean "MPLS TTL" and using "IP TTL" for the TTL value in the IP header.

### 1.2. Structure of This Document

The body of this memo contains four main parts: motivation, MPLS echo request/reply packet format, LSP ping operation, and a reliable return path. It is suggested that first-time readers skip the actual packet formats and read the Theory of Operation first; the document is structured the way it is to avoid forward references.

### 1.3. Contributors

A mechanism used to detect data plane failures in Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) was originally published as RFC 4379 in February 2006. It was produced by the MPLS Working Group of the IETF and was jointly authored by Kireeti Kompella and George Swallow.

The following made vital contributions to all aspects of the original RFC 4379, and much of the material came out of debate and discussion among this group.

Ronald P. Bonica, Juniper Networks, Inc.  
Dave Cooper, Global Crossing  
Ping Pan, Hammerhead Systems  
Nischal Sheth, Juniper Networks, Inc.  
Sanjay Wadhwa, Juniper Networks, Inc.

#### 1.4. Scope of RFC4379bis work

The goal of this document is to take LSP Ping to an Internet Standard.

[RFC4379] defines the basic mechanism for MPLS LSP validation that can be used for fault detection and isolation. The scope of this document also is to address various updates to MPLS LSP Ping, including:

1. Updates to all references and citations. Obsoleted RFCs 2434, 2030, and 3036 are respectively replaced with RFCs 5226, 5905, and 5036. Additionally, these three documents published as RFCs: RFCs 4447, 5085, and 4761.
2. Incorporate all outstanding Errata. These include Erratum with IDs: 108, 1418, 1714, 1786, 3399, 742, and 2978.
3. Replace EXP with Traffic Class (TC), based on the update from RFC 5462.
4. Incorporate the updates from RFC 6829, adding the PW FECs advertised over IPv6.
5. Incorporate the updates from RFC 7506, adding IPv6 Router Alert Option for MPLS OAM.

#### 1.5. ToDo

This section should be empty, and removed, prior to publication. ToDos:

1. Evaluation of which of the RFCs that updated RFC 4379 need to be incorporated into this 4379bis document. Specifically, these RFCs updated RFC 4379: 6424, 6425, 6426 and 7537. RFCs that updated RFC 4379 and are incorporated into this 4379bis, will be Obsoleted by 4379bis.
2. Review IANA Allocations

#### 2. Motivation

When an LSP fails to deliver user traffic, the failure cannot always be detected by the MPLS control plane. There is a need to provide a tool that would enable users to detect such traffic "black holes" or misrouting within a reasonable period of time, and a mechanism to isolate faults.

In this document, we describe a mechanism that accomplishes these goals. This mechanism is modeled after the ping/traceroute paradigm: ping (ICMP echo request [RFC0792]) is used for connectivity checks, and traceroute is used for hop-by-hop fault localization as well as

path tracing. This document specifies a "ping" mode and a "traceroute" mode for testing MPLS LSPs.

The basic idea is to verify that packets that belong to a particular Forwarding Equivalence Class (FEC) actually end their MPLS path on a Label Switching Router (LSR) that is an egress for that FEC. This document proposes that this test be carried out by sending a packet (called an "MPLS echo request") along the same data path as other packets belonging to this FEC. An MPLS echo request also carries information about the FEC whose MPLS path is being verified. This echo request is forwarded just like any other packet belonging to that FEC. In "ping" mode (basic connectivity check), the packet should reach the end of the path, at which point it is sent to the control plane of the egress LSR, which then verifies whether it is indeed an egress for the FEC. In "traceroute" mode (fault isolation), the packet is sent to the control plane of each transit LSR, which performs various checks that it is indeed a transit LSR for this path; this LSR also returns further information that helps check the control plane against the data plane, i.e., that forwarding matches what the routing protocols determined as the path.

One way these tools can be used is to periodically ping an FEC to ensure connectivity. If the ping fails, one can then initiate a traceroute to determine where the fault lies. One can also periodically traceroute FECs to verify that forwarding matches the control plane; however, this places a greater burden on transit LSRs and thus should be used with caution.

#### 2.1. Use of Address Range 127/8

As described above, LSP ping is intended as a diagnostic tool. It is intended to enable providers of an MPLS-based service to isolate network faults. In particular, LSP ping needs to diagnose situations where the control and data planes are out of sync. It performs this by routing an MPLS echo request packet based solely on its label stack. That is, the IP destination address is never used in a forwarding decision. In fact, the sender of an MPLS echo request packet may not know, a priori, the address of the router at the end of the LSP.

Providers of MPLS-based services also need the ability to trace all of the possible paths that an LSP may take. Since most MPLS services are based on IP unicast forwarding, these paths are subject to equal-cost multi-path (ECMP) load sharing.

This leads to the following requirements:

1. Although the LSP in question may be broken in unknown ways, the likelihood of a diagnostic packet being delivered to a user of an MPLS service MUST be held to an absolute minimum.
2. If an LSP is broken in such a way that it prematurely terminates, the diagnostic packet MUST NOT be IP forwarded.
3. A means of varying the diagnostic packets such that they exercise all ECMP paths is thus REQUIRED.

Clearly, using general unicast addresses satisfies neither of the first two requirements. A number of other options for addresses were considered, including a portion of the private address space (as determined by the network operator) and the newly designated IPv4 link local addresses. Use of the private address space was deemed ineffective since the leading MPLS-based service is an IPv4 Virtual Private Network (VPN). VPNs often use private addresses.

The IPv4 link local addresses are more attractive in that the scope over which they can be forwarded is limited. However, if one were to use an address from this range, it would still be possible for the first recipient of a diagnostic packet that "escaped" from a broken LSP to have that address assigned to the interface on which it arrived and thus could mistakenly receive such a packet. Furthermore, the IPv4 link local address range has only recently been allocated. Many deployed routers would forward a packet with an address from that range toward the default route.

The 127/8 range for IPv4 and that same range embedded in as IPv4-mapped IPv6 addresses for IPv6 was chosen for a number of reasons.

RFC 1122 allocates the 127/8 as "Internal host loopback address" and states: "Addresses of this form MUST NOT appear outside a host." Thus, the default behavior of hosts is to discard such packets. This helps to ensure that if a diagnostic packet is misdirected to a host, it will be silently discarded.

RFC 1812 [RFC1812] states:

A router SHOULD NOT forward, except over a loopback interface, any packet that has a destination address on network 127. A router MAY have a switch that allows the network manager to disable these checks. If such a switch is provided, it MUST default to performing the checks.

This helps to ensure that diagnostic packets are never IP forwarded.

The 127/8 address range provides 16M addresses allowing wide flexibility in varying addresses to exercise ECMP paths. Finally, as an implementation optimization, the 127/8 provides an easy means of identifying possible LSP packets.

## 2.2. Router Alert Option

This document requires the use of the Router Alert Option (RAO) set in IP header in order to have the transit node process the MPLS OAM payload.

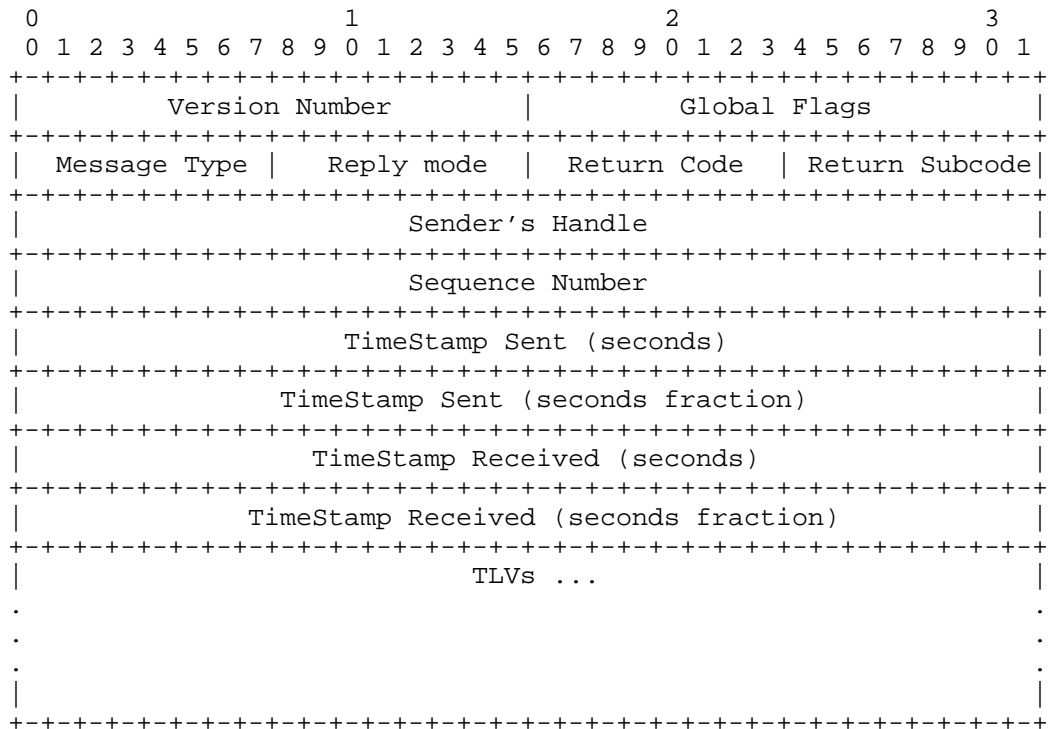
[RFC2113] defines a generic Option Value 0x0 for IPv4 RAO that alerts transit router to examine the IPv4 packet. [RFC7506] defines MPLS OAM Option Value 69 for IPv6 RAO to alert transit routers to examine the IPv6 packet more closely for MPLS OAM purposes.

The use of the Router Alert IP Option in this document is as follows:

In case of an IPv4 header, the generic IPv4 RAO value 0x0 [RFC2113] SHOULD be used. In case of an IPv6 header, the IPv6 RAO value (69) for MPLS OAM [RFC7506] MUST be used.

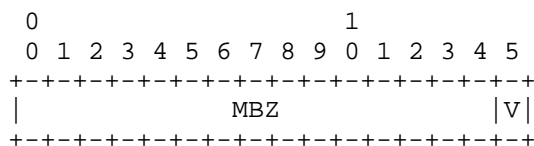
## 3. Packet Format

An MPLS echo request is a (possibly labeled) IPv4 or IPv6 UDP packet; the contents of the UDP packet have the following format:



The Version Number is currently 1. (Note: the version number is to be incremented whenever a change is made that affects the ability of an implementation to correctly parse or process an MPLS echo request/reply. These changes include any syntactic or semantic changes made to any of the fixed fields, or to any Type-Length-Value (TLV) or sub-TLV assignment or format that is defined at a certain version number. The version number may not need to be changed if an optional TLV or sub-TLV is added.)

The Global Flags field is a bit vector with the following format:



One flag is defined for now, the V bit; the rest MUST be set to zero when sending and ignored on receipt.

The V (Validate FEC Stack) flag is set to 1 if the sender wants the receiver to perform FEC Stack validation; if V is 0, the choice is left to the receiver.

The Message Type is one of the following:

Value	Meaning
-----	-----
1	MPLS echo request
2	MPLS echo reply

The Reply Mode can take one of the following values:

Value	Meaning
-----	-----
1	Do not reply
2	Reply via an IPv4/IPv6 UDP packet
3	Reply via an IPv4/IPv6 UDP packet with Router Alert
4	Reply via application level control channel

An MPLS echo request with 1 (Do not reply) in the Reply Mode field may be used for one-way connectivity tests; the receiving router may log gaps in the Sequence Numbers and/or maintain delay/jitter statistics. An MPLS echo request would normally have 2 (Reply via an IPv4/IPv6 UDP packet) in the Reply Mode field. If the normal IP return path is deemed unreliable, one may use 3 (Reply via an IPv4/IPv6 UDP packet with Router Alert). Note that this requires that all intermediate routers understand and know how to forward MPLS echo replies. The echo reply uses the same IP version number as the received echo request, i.e., an IPv4 encapsulated echo reply is sent in response to an IPv4 encapsulated echo request.

Some applications support an IP control channel. One such example is the associated control channel defined in Virtual Circuit Connectivity Verification (VCCV) [RFC5085]. Any application that supports an IP control channel between its control entities may set the Reply Mode to 4 (Reply via application level control channel) to ensure that replies use that same channel. Further definition of this codepoint is application specific and thus beyond the scope of this document.

Return Codes and Subcodes are described in the next section.

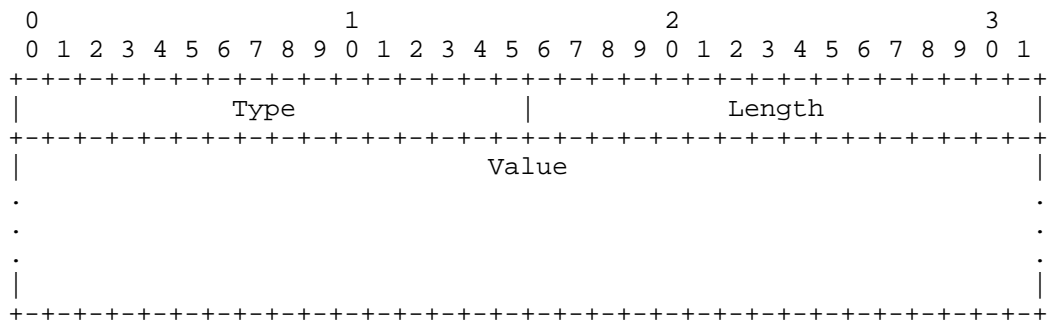
The Sender's Handle is filled in by the sender, and returned unchanged by the receiver in the echo reply (if any). There are no semantics associated with this handle, although a sender may find this useful for matching up requests with replies.



The Sequence Number is assigned by the sender of the MPLS echo request and can be (for example) used to detect missed replies.

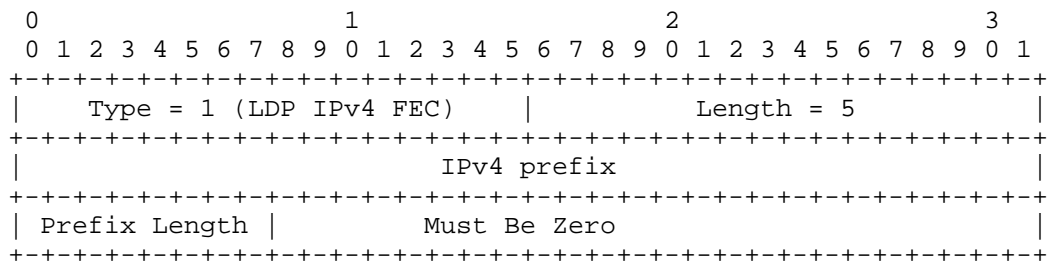
The TimeStamp Sent is the time-of-day (according to the sender's clock) in NTP format [RFC5905] when the MPLS echo request is sent. The TimeStamp Received in an echo reply is the time-of-day (according to the receiver's clock) in NTP format that the corresponding echo request was received.

TLVs (Type-Length-Value tuples) have the following format:

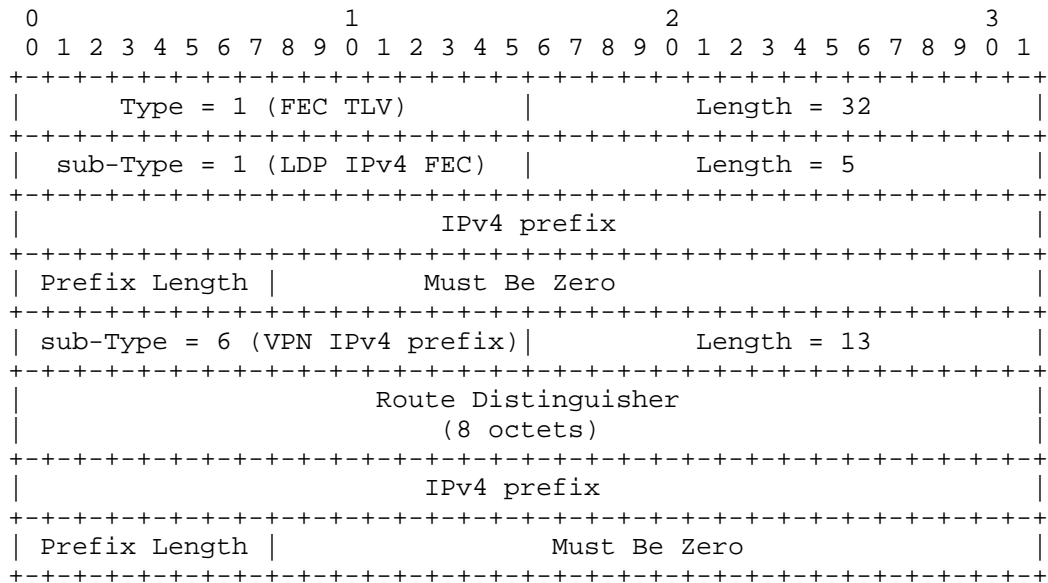


Types are defined below; Length is the length of the Value field in octets. The Value field depends on the Type; it is zero padded to align to a 4-octet boundary. TLVs may be nested within other TLVs, in which case the nested TLVs are called sub-TLVs. Sub-TLVs have independent types and MUST also be 4-octet aligned.

Two examples follow. The Label Distribution Protocol (LDP) IPv4 FEC sub-TLV has the following format:



The Length for this TLV is 5. A Target FEC Stack TLV that contains an LDP IPv4 FEC sub-TLV and a VPN IPv4 prefix sub-TLV has the following format:



A description of the Types and Values of the top-level TLVs for LSP ping are given below:

Type #	Value Field
-----	-----
1	Target FEC Stack
2	Downstream Mapping
3	Pad
4	Not Assigned
5	Vendor Enterprise Number
6	Not Assigned
7	Interface and Label Stack
8	Not Assigned
9	Errored TLVs
10	Reply TOS Byte

Types less than 32768 (i.e., with the high-order bit equal to 0) are mandatory TLVs that MUST either be supported by an implementation or result in the return code of 2 ("One or more of the TLVs was not understood") being sent in the echo response.

Types greater than or equal to 32768 (i.e., with the high-order bit equal to 1) are optional TLVs that SHOULD be ignored if the implementation does not understand or support them.

### 3.1. Return Codes

The Return Code is set to zero by the sender. The receiver can set it to one of the values listed below. The notation <RSC> refers to the Return Subcode. This field is filled in with the stack-depth for those codes that specify that. For all other codes, the Return Subcode MUST be set to zero.

Value	Meaning
-----	-----
0	No return code
1	Malformed echo request received
2	One or more of the TLVs was not understood
3	Replying router is an egress for the FEC at stack-depth <RSC>
4	Replying router has no mapping for the FEC at stack-depth <RSC>
5	Downstream Mapping Mismatch (See Note 1)
6	Upstream Interface Index Unknown (See Note 1)
7	Reserved
8	Label switched at stack-depth <RSC>
9	Label switched but no MPLS forwarding at stack-depth <RSC>
10	Mapping for this FEC is not the given label at stack-depth <RSC>
11	No label entry at stack-depth <RSC>
12	Protocol not associated with interface at FEC stack-depth <RSC>
13	Premature termination of ping due to label stack shrinking to a single label

#### Note 1

The Return Subcode contains the point in the label stack where processing was terminated. If the RSC is 0, no labels were processed. Otherwise the packet would have been label switched at depth RSC.

### 3.2. Target FEC Stack

A Target FEC Stack is a list of sub-TLVs. The number of elements is determined by looking at the sub-TLV length fields.

Sub-Type	Length	Value Field
-----	-----	-----
1	5	LDP IPv4 prefix
2	17	LDP IPv6 prefix
3	20	RSVP IPv4 LSP
4	56	RSVP IPv6 LSP
5		Not Assigned
6	13	VPN IPv4 prefix
7	25	VPN IPv6 prefix
8	14	L2 VPN endpoint
9	10	"FEC 128" Pseudowire - IPv4 (deprecated)
10	14	"FEC 128" Pseudowire - IPv4
11	16+	"FEC 129" Pseudowire - IPv4
12	5	BGP labeled IPv4 prefix
13	17	BGP labeled IPv6 prefix
14	5	Generic IPv4 prefix
15	17	Generic IPv6 prefix
16	4	Nil FEC
24	38	"FEC 128" Pseudowire - IPv6
25	40+	"FEC 129" Pseudowire - IPv6

Other FEC Types will be defined as needed.

Note that this TLV defines a stack of FECs, the first FEC element corresponding to the top of the label stack, etc.

An MPLS echo request MUST have a Target FEC Stack that describes the FEC Stack being tested. For example, if an LSR X has an LDP mapping [RFC5036] for 192.168.1.1 (say, label 1001), then to verify that label 1001 does indeed reach an egress LSR that announced this prefix via LDP, X can send an MPLS echo request with an FEC Stack TLV with one FEC in it, namely, of type LDP IPv4 prefix, with prefix 192.168.1.1/32, and send the echo request with a label of 1001.

Say LSR X wanted to verify that a label stack of <1001, 23456> is the right label stack to use to reach a VPN IPv4 prefix [see section 3.2.5] of 10/8 in VPN foo. Say further that LSR Y with loopback address 192.168.1.1 announced prefix 10/8 with Route Distinguisher RD-foo-Y (which may in general be different from the Route Distinguisher that LSR X uses in its own advertisements for VPN foo), label 23456 and BGP next hop 192.168.1.1 [RFC4271]. Finally, suppose that LSR X receives a label binding of 1001 for 192.168.1.1 via LDP. X has two choices in sending an MPLS echo request: X can send an MPLS echo request with an FEC Stack TLV with a single FEC of type VPN IPv4 prefix with a prefix of 10/8 and a Route Distinguisher of RD-foo-Y. Alternatively, X can send an FEC Stack TLV with two FECs, the first of type LDP IPv4 with a prefix of 192.168.1.1/32 and the second of type of IP VPN with a prefix 10/8 with Route Distinguisher of RD-foo-

Y. In either case, the MPLS echo request would have a label stack of <1001, 23456>. (Note: in this example, 1001 is the "outer" label and 23456 is the "inner" label.)

### 3.2.1. LDP IPv4 Prefix

The IPv4 Prefix FEC is defined in [RFC5036]. When an LDP IPv4 prefix is encoded in a label stack, the following format is used. The value consists of 4 octets of an IPv4 prefix followed by 1 octet of prefix length in bits; the format is given below. The IPv4 prefix is in network byte order; if the prefix is shorter than 32 bits, trailing bits SHOULD be set to zero. See [RFC5036] for an example of a Mapping for an IPv4 FEC.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     IPv4 prefix                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Prefix Length |                               Must Be Zero                               |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

### 3.2.2. LDP IPv6 Prefix

The IPv6 Prefix FEC is defined in [RFC5036]. When an LDP IPv6 prefix is encoded in a label stack, the following format is used. The value consists of 16 octets of an IPv6 prefix followed by 1 octet of prefix length in bits; the format is given below. The IPv6 prefix is in network byte order; if the prefix is shorter than 128 bits, the trailing bits SHOULD be set to zero. See [RFC5036] for an example of a Mapping for an IPv6 FEC.

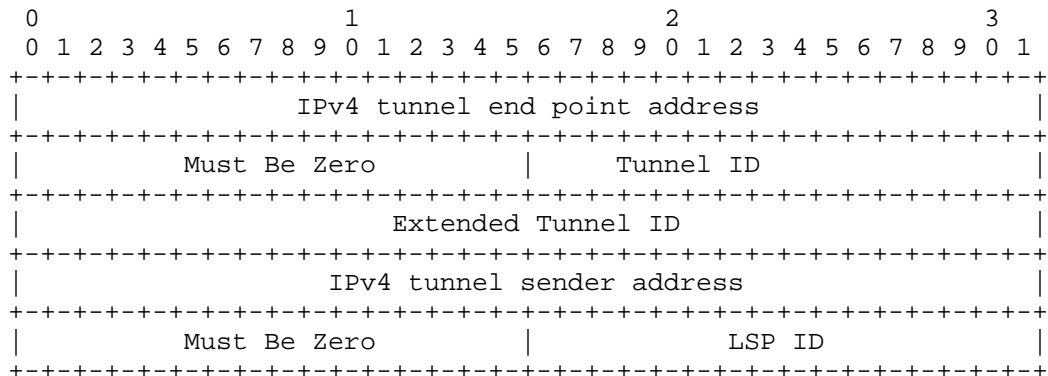
```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     IPv6 prefix                                     |
|                                     (16 octets)                                    |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Prefix Length |                               Must Be Zero                               |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

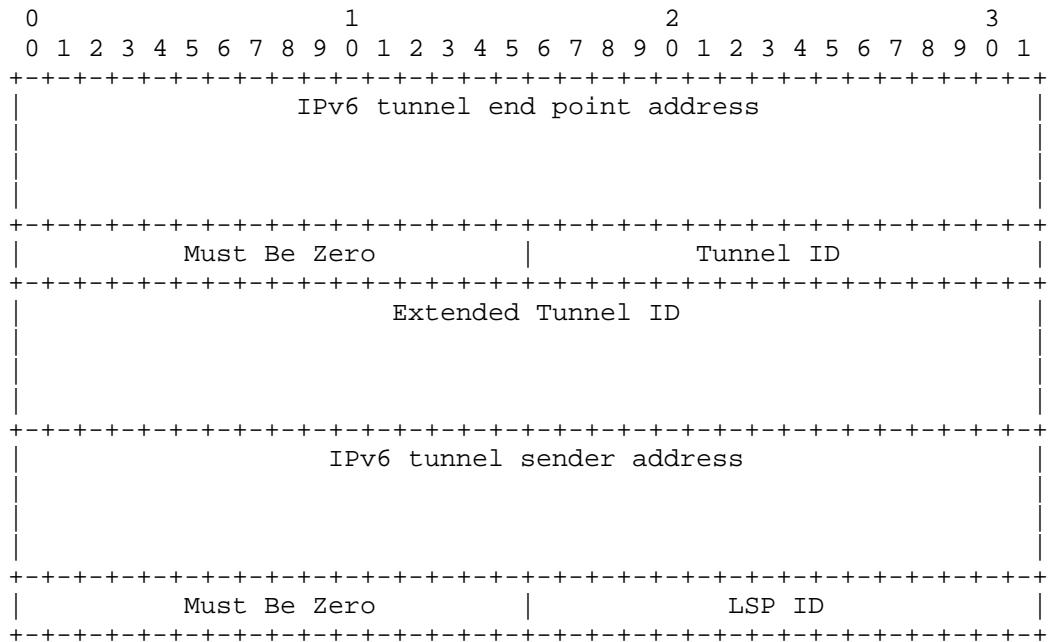
### 3.2.3. RSVP IPv4 LSP

The value has the format below. The value fields are taken from RFC 3209, sections 4.6.1.1 and 4.6.2.1. See [RFC3209].



#### 3.2.4. RSVP IPv6 LSP

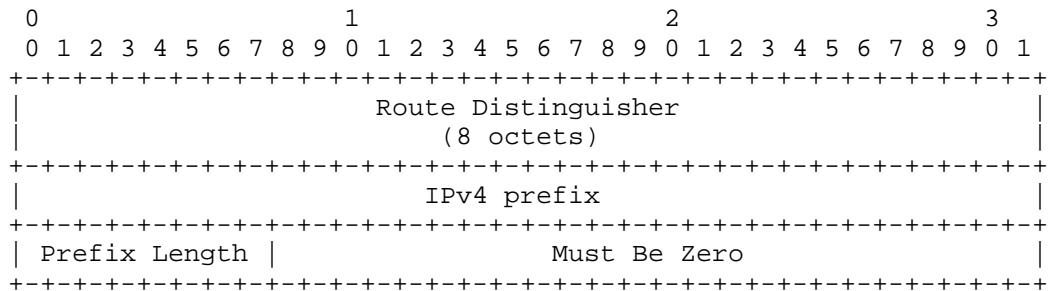
The value has the format below. The value fields are taken from RFC 3209, sections 4.6.1.2 and 4.6.2.2. See [RFC3209].



#### 3.2.5. VPN IPv4 Prefix

VPN-IPv4 Network Layer Routing Information (NLRI) is defined in [RFC4365]. This document uses the term VPN IPv4 prefix for a VPN-IPv4 NLRI that has been advertised with an MPLS label in BGP. See [RFC3107].

When a VPN IPv4 prefix is encoded in a label stack, the following format is used. The value field consists of the Route Distinguisher advertised with the VPN IPv4 prefix, the IPv4 prefix (with trailing 0 bits to make 32 bits in all), and a prefix length, as follows:

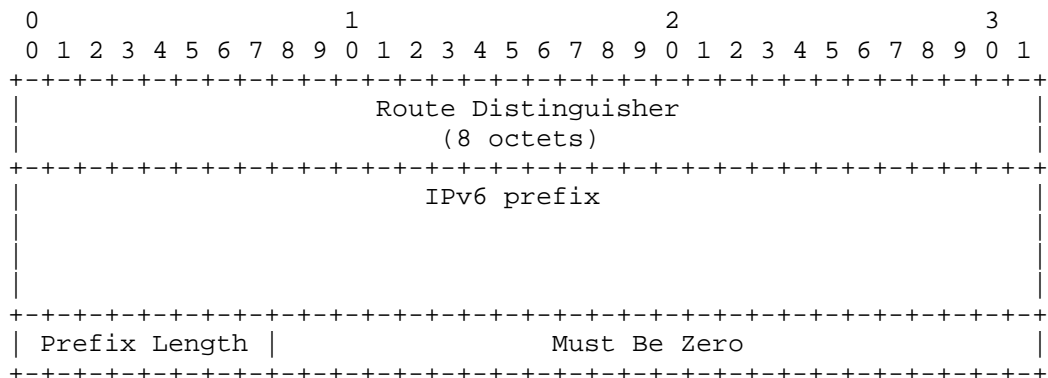


The Route Distinguisher (RD) is an 8-octet identifier; it does not contain any inherent information. The purpose of the RD is solely to allow one to create distinct routes to a common IPv4 address prefix. The encoding of the RD is not important here. When matching this field to the local FEC information, it is treated as an opaque value.

### 3.2.6. VPN IPv6 Prefix

VPN-IPv6 Network Layer Routing Information (NLRI) is defined in [RFC4365]. This document uses the term VPN IPv6 prefix for a VPN-IPv6 NLRI that has been advertised with an MPLS label in BGP. See [RFC3107].

When a VPN IPv6 prefix is encoded in a label stack, the following format is used. The value field consists of the Route Distinguisher advertised with the VPN IPv6 prefix, the IPv6 prefix (with trailing 0 bits to make 128 bits in all), and a prefix length, as follows:

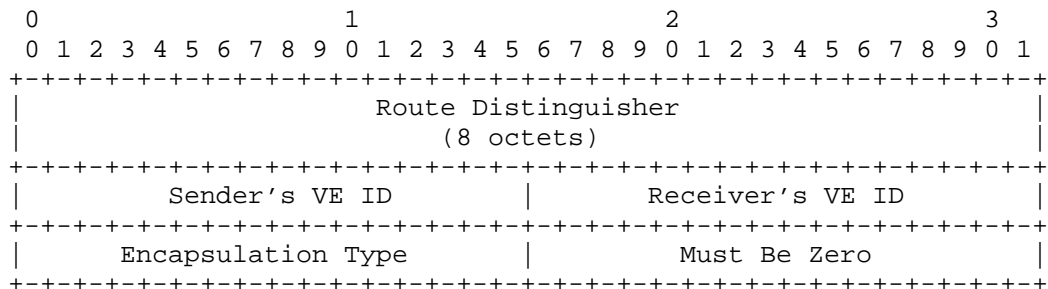


The Route Distinguisher is identical to the VPN IPv4 Prefix RD, except that it functions here to allow the creation of distinct routes to IPv6 prefixes. See section 3.2.5. When matching this field to local FEC information, it is treated as an opaque value.

### 3.2.7. L2 VPN Endpoint

VPLS stands for Virtual Private LAN Service. The terms VPLS BGP NLRI and VE ID (VPLS Edge Identifier) are defined in [RFC4761]. This document uses the simpler term L2 VPN endpoint when referring to a VPLS BGP NLRI. The Route Distinguisher is an 8-octet identifier used to distinguish information about various L2 VPNs advertised by a node. The VE ID is a 2-octet identifier used to identify a particular node that serves as the service attachment point within a VPLS. The structure of these two identifiers is unimportant here; when matching these fields to local FEC information, they are treated as opaque values. The encapsulation type is identical to the PW Type in section 3.2.8 below.

When an L2 VPN endpoint is encoded in a label stack, the following format is used. The value field consists of a Route Distinguisher (8 octets), the sender (of the ping)'s VE ID (2 octets), the receiver's VE ID (2 octets), and an encapsulation type (2 octets), formatted as follows:



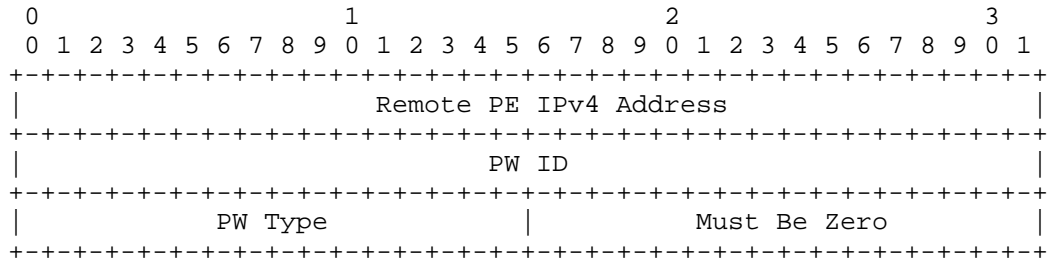
### 3.2.8. FEC 128 Pseudowire - IPv4 (Deprecated)

FEC 128 (0x80) is defined in [RFC4447], as are the terms PW ID (Pseudowire ID) and PW Type (Pseudowire Type). A PW ID is a non-zero 32-bit connection ID. The PW Type is a 15-bit number indicating the encapsulation type. It is carried right justified in the field below termed encapsulation type with the high-order bit set to zero. Both of these fields are treated in this protocol as opaque values.

When an FEC 128 is encoded in a label stack, the following format is used. The value field consists of the remote PE IPv4 address (the



destination address of the targeted LDP session), the PW ID, and the encapsulation type as follows:



This FEC is deprecated and is retained only for backward compatibility. Implementations of LSP ping SHOULD accept and process this TLV, but SHOULD send LSP ping echo requests with the new TLV (see next section), unless explicitly configured to use the old TLV.

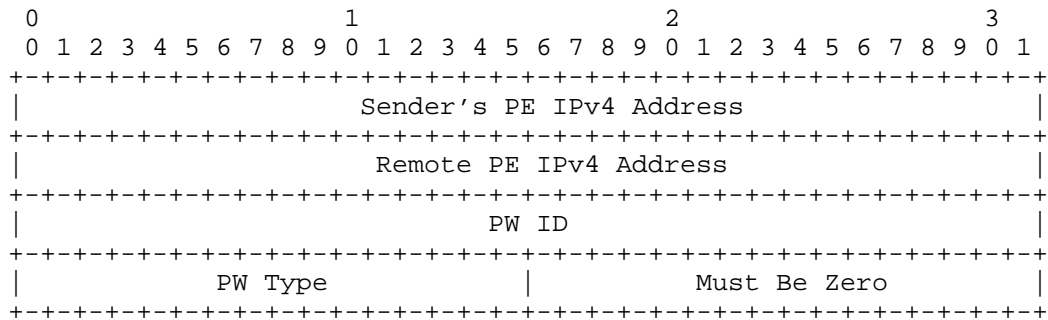
An LSR receiving this TLV SHOULD use the source IP address of the LSP echo request to infer the sender's PE address.

### 3.2.9. FEC 128 Pseudowire - IPv4 (Current)

FEC 128 (0x80) is defined in [RFC4447], as are the terms PW ID (Pseudowire ID) and PW Type (Pseudowire Type). A PW ID is a non-zero 32-bit connection ID. The PW Type is a 15-bit number indicating the encapsulation type. It is carried right justified in the field below termed encapsulation type with the high-order bit set to zero.

Both of these fields are treated in this protocol as opaque values. When matching these field to the local FEC information, the match MUST be exact.

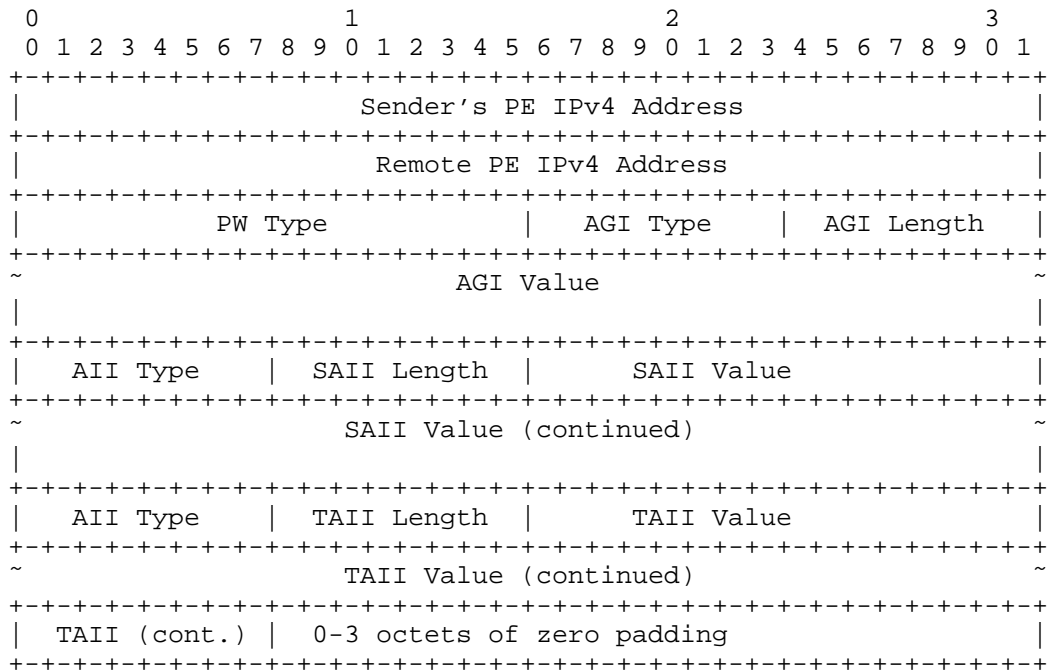
When an FEC 128 is encoded in a label stack, the following format is used. The value field consists of the sender's PE IPv4 address (the source address of the targeted LDP session), the remote PE IPv4 address (the destination address of the targeted LDP session), the PW ID, and the encapsulation type as follows:



### 3.2.10. FEC 129 Pseudowire - IPv4

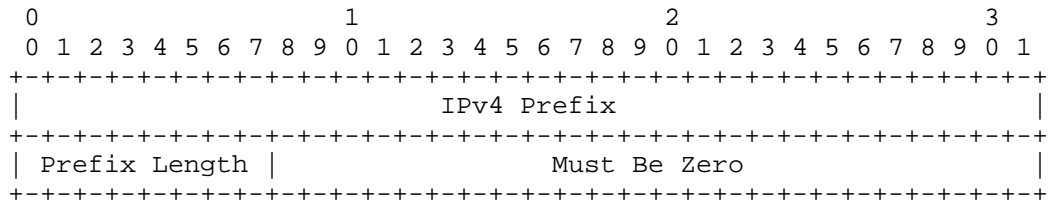
FEC 129 (0x81) and the terms PW Type, Attachment Group Identifier (AGI), Attachment Group Identifier Type (AGI Type), Attachment Individual Identifier Type (AII Type), Source Attachment Individual Identifier (SAII), and Target Attachment Individual Identifier (TAII) are defined in [RFC4447]. The PW Type is a 15-bit number indicating the encapsulation type. It is carried right justified in the field below PW Type with the high-order bit set to zero. All the other fields are treated as opaque values and copied directly from the FEC 129 format. All of these values together uniquely define the FEC within the scope of the LDP session identified by the source and remote PE IPv4 addresses.

When an FEC 129 is encoded in a label stack, the following format is used. The Length of this TLV is 16 + AGI length + SAII length + TAII length. Padding is used to make the total length a multiple of 4; the length of the padding is not included in the Length field.



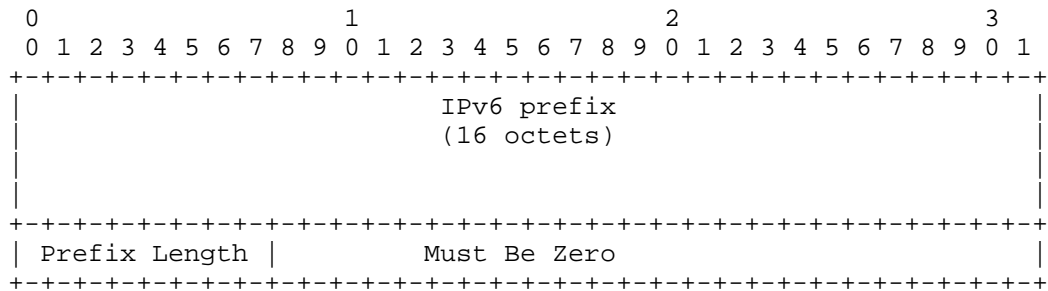
### 3.2.11. BGP Labeled IPv4 Prefix

BGP labeled IPv4 prefixes are defined in [RFC3107]. When a BGP labeled IPv4 prefix is encoded in a label stack, the following format is used. The value field consists the IPv4 prefix (with trailing 0 bits to make 32 bits in all), and the prefix length, as follows:



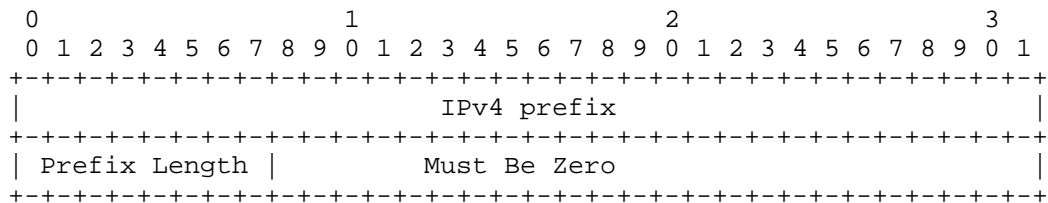
### 3.2.12. BGP Labeled IPv6 Prefix

BGP labeled IPv6 prefixes are defined in [RFC3107]. When a BGP labeled IPv6 prefix is encoded in a label stack, the following format is used. The value consists of 16 octets of an IPv6 prefix followed by 1 octet of prefix length in bits; the format is given below. The IPv6 prefix is in network byte order; if the prefix is shorter than 128 bits, the trailing bits SHOULD be set to zero.



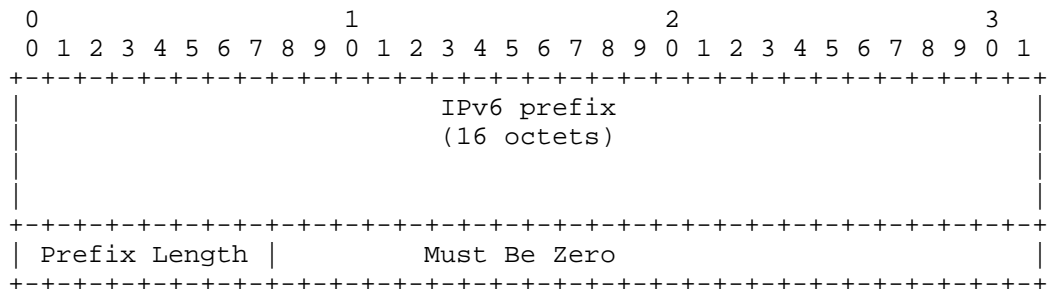
### 3.2.13. Generic IPv4 Prefix

The value consists of 4 octets of an IPv4 prefix followed by 1 octet of prefix length in bits; the format is given below. The IPv4 prefix is in network byte order; if the prefix is shorter than 32 bits, trailing bits SHOULD be set to zero. This FEC is used if the protocol advertising the label is unknown or may change during the course of the LSP. An example is an inter-AS LSP that may be signaled by LDP in one Autonomous System (AS), by RSVP-TE [RFC3209] in another AS, and by BGP between the ASes, such as is common for inter-AS VPNs.



### 3.2.14. Generic IPv6 Prefix

The value consists of 16 octets of an IPv6 prefix followed by 1 octet of prefix length in bits; the format is given below. The IPv6 prefix is in network byte order; if the prefix is shorter than 128 bits, the trailing bits SHOULD be set to zero.



## 3.2.15. Nil FEC

At times, labels from the reserved range, e.g., Router Alert and Explicit-null, may be added to the label stack for various diagnostic purposes such as influencing load-balancing. These labels may have no explicit FEC associated with them. The Nil FEC Stack is defined to allow a Target FEC Stack sub-TLV to be added to the Target FEC Stack to account for such labels so that proper validation can still be performed.

The Length is 4. Labels are 20-bit values treated as numbers.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     |                                     |
|                                     |                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Label is the actual label value inserted in the label stack; the MBZ fields MUST be zero when sent and ignored on receipt.

## 3.2.16. FEC 128 Pseudowire - IPv6

The FEC 128 Pseudowire IPv6 sub-TLV has a structure consistent with the FEC 128 Pseudowire IPv4 sub-TLV as described in Section 3.2.9. The value field consists of the Sender's PE IPv6 address (the source address of the targeted LDP session), the remote PE IPv6 address (the destination address of the targeted LDP session), the PW ID, and the encapsulation type as follows:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
~                                     ~
~                                     ~
+-----+-----+-----+-----+-----+-----+-----+-----+
~                                     ~
~                                     ~
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     |                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     |                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Sender's PE IPv6 Address: The source IP address of the target IPv6 LDP session. 16 octets.

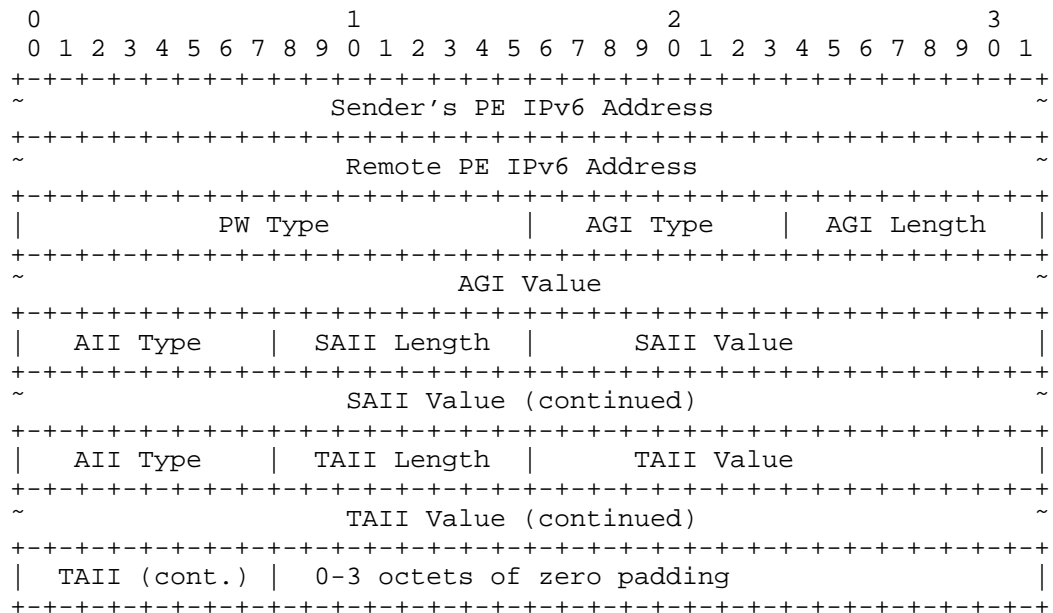
Remote PE IPv6 Address: The destination IP address of the target IPv6 LDP session. 16 octets.

PW ID: Same as FEC 128 Pseudowire IPv4 in Section 3.2.9.

PW Type: Same as FEC 128 Pseudowire IPv4 in Section 3.2.9.

### 3.2.17. FEC 129 Pseudowire - IPv6

The FEC 129 Pseudowire IPv6 sub-TLV has a structure consistent with the FEC 129 Pseudowire IPv4 sub-TLV as described in Section 3.2.10. When an FEC 129 is encoded in a label stack, the following format is used. The length of this TLV is 40 + AGI (Attachment Group Identifier) length + SAII (Source Attachment Individual Identifier) length + TAI (Target Attachment Individual Identifier) length. Padding is used to make the total length a multiple of 4; the length of the padding is not included in the Length field.



Sender's PE IPv6 Address: The source IP address of the target IPv6 LDP session. 16 octets.

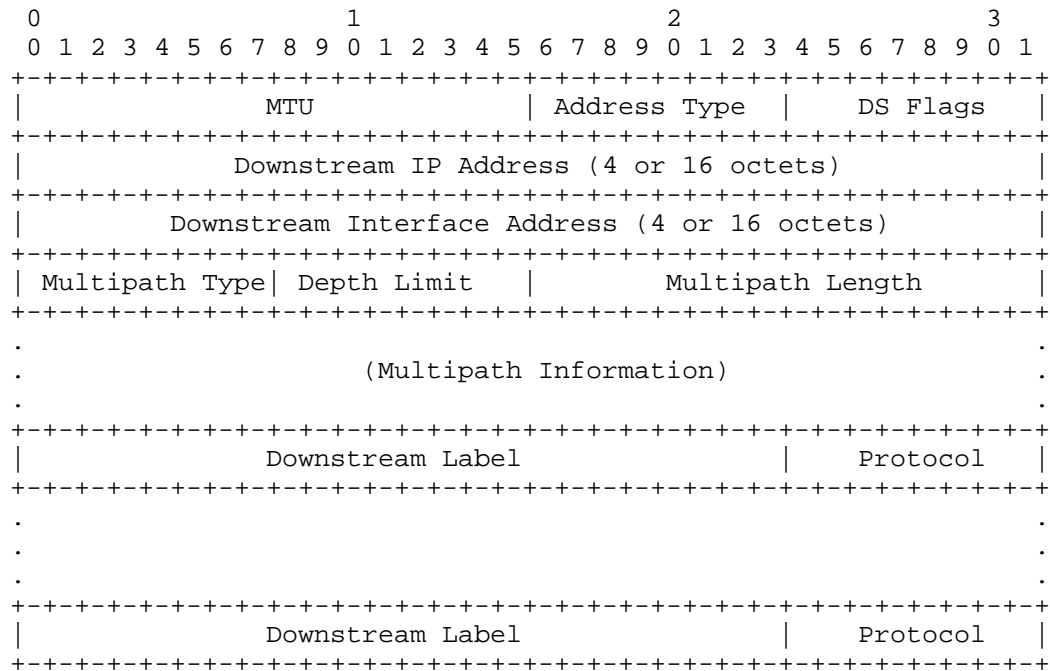
Remote PE IPv6 Address: The destination IP address of the target IPv6 LDP session. 16 octets.

The other fields are the same as FEC 129 Pseudowire IPv4 in Section 3.2.10.

### 3.3. Downstream Mapping

The Downstream Mapping object is a TLV that MAY be included in an echo request message. Only one Downstream Mapping object may appear in an echo request. The presence of a Downstream Mapping object is a request that Downstream Mapping objects be included in the echo reply. If the replying router is the destination of the FEC, then a Downstream Mapping TLV SHOULD NOT be included in the echo reply. Otherwise the replying router SHOULD include a Downstream Mapping object for each interface over which this FEC could be forwarded. For a more precise definition of the notion of "downstream", see section 3.3.2, "Downstream Router and Interface".

The Length is  $K + M + 4*N$  octets, where  $M$  is the Multipath Length, and  $N$  is the number of Downstream Labels. Values for  $K$  are found in the description of Address Type below. The Value field of a Downstream Mapping has the following format:



#### Maximum Transmission Unit (MTU)

The MTU is the size in octets of the largest MPLS frame (including label stack) that fits on the interface to the Downstream LSR.

#### Address Type

The Address Type indicates if the interface is numbered or unnumbered. It also determines the length of the Downstream IP Address and Downstream Interface fields. The resulting total for the initial part of the TLV is listed in the table below as "K Octets". The Address Type is set to one of the following values:

Type #	Address Type	K Octets
-----	-----	-----
1	IPv4 Numbered	16
2	IPv4 Unnumbered	16
3	IPv6 Numbered	40
4	IPv6 Unnumbered	28

#### DS Flags

The DS Flags field is a bit vector with the following format:

```

  0 1 2 3 4 5 6 7
+-----+
| Rsvd(MBZ) | I | N |
+-----+
```

Two flags are defined currently, I and N. The remaining flags MUST be set to zero when sending and ignored on receipt.

#### Flag Name and Meaning

-----

I Interface and Label Stack Object Request

When this flag is set, it indicates that the replying router SHOULD include an Interface and Label Stack Object in the echo reply message.

N Treat as a Non-IP Packet

Echo request messages will be used to diagnose non-IP flows. However, these messages are carried in IP packets. For a router that alters its ECMP algorithm based on the FEC or deep packet examination, this flag requests that the router treat this as it would if the determination of an IP payload had failed.

#### Downstream IP Address and Downstream Interface Address

IPv4 addresses and interface indices are encoded in 4 octets; IPv6 addresses are encoded in 16 octets.



If the interface to the downstream LSR is numbered, then the Address Type MUST be set to IPv4 or IPv6, the Downstream IP Address MUST be set to either the downstream LSR's Router ID or the interface address of the downstream LSR, and the Downstream Interface Address MUST be set to the downstream LSR's interface address.

If the interface to the downstream LSR is unnumbered, the Address Type MUST be IPv4 Unnumbered or IPv6 Unnumbered, the Downstream IP Address MUST be the downstream LSR's Router ID, and the Downstream Interface Address MUST be set to the index assigned by the upstream LSR to the interface.

If an LSR does not know the IP address of its neighbor, then it MUST set the Address Type to either IPv4 Unnumbered or IPv6 Unnumbered. For IPv4, it must set the Downstream IP Address to 127.0.0.1; for IPv6 the address is set to 0::1. In both cases, the interface index MUST be set to 0. If an LSR receives an Echo Request packet with either of these addresses in the Downstream IP Address field, this indicates that it MUST bypass interface verification but continue with label validation.

If the originator of an Echo Request packet wishes to obtain Downstream Mapping information but does not know the expected label stack, then it SHOULD set the Address Type to either IPv4 Unnumbered or IPv6 Unnumbered. For IPv4, it MUST set the Downstream IP Address to 224.0.0.2; for IPv6 the address MUST be set to FF02::2. In both cases, the interface index MUST be set to 0. If an LSR receives an Echo Request packet with the all-routers multicast address, then this indicates that it MUST bypass both interface and label stack validation, but return Downstream Mapping TLVs using the information provided.

#### Multipath Type

The following Multipath Types are defined:

Key	Type	Multipath Information
---	-----	-----
0	no multipath	Empty (Multipath Length = 0)
2	IP address	IP addresses
4	IP address range	low/high address pairs
8	Bit-masked IP address set	IP address prefix and bit mask
9	Bit-masked label set	Label prefix and bit mask

Type 0 indicates that all packets will be forwarded out this one interface.

Types 2, 4, 8, and 9 specify that the supplied Multipath Information will serve to exercise this path.

#### Depth Limit

The Depth Limit is applicable only to a label stack and is the maximum number of labels considered in the hash; this SHOULD be set to zero if unspecified or unlimited.

#### Multipath Length

The length in octets of the Multipath Information.

#### Multipath Information

Address or label values encoded according to the Multipath Type. See the next section below for encoding details.

#### Downstream Label(s)

The set of labels in the label stack as it would have appeared if this router were forwarding the packet through this interface. Any Implicit Null labels are explicitly included. Labels are treated as numbers, i.e., they are right justified in the field.

A Downstream Label is 24 bits, in the same format as an MPLS label minus the TTL field, i.e., the MSBit of the label is bit 0, the LSBit is bit 19, the Traffic Class (TC) bits are bits 20-22, and bit 23 is the S bit. The replying router SHOULD fill in the TC and S bits; the LSR receiving the echo reply MAY choose to ignore these bits. Protocol

The Protocol is taken from the following table:

Protocol #	Signaling Protocol
-----	-----
0	Unknown
1	Static
2	BGP
3	LDP
4	RSVP-TE

#### 3.3.1. Multipath Information Encoding

The Multipath Information encodes labels or addresses that will exercise this path. The Multipath Information depends on the Multipath Type. The contents of the field are shown in the table above. IPv4 addresses are drawn from the range 127/8; IPv6 addresses

are drawn from the range 0:0:0:0:0:FFFF:7F00/104. Labels are treated as numbers, i.e., they are right justified in the field. For Type 4, ranges indicated by Address pairs MUST NOT overlap and MUST be in ascending sequence.

Type 8 allows a more dense encoding of IP addresses. The IP prefix is formatted as a base IP address with the non-prefix low-order bits set to zero. The maximum prefix length is 27. Following the prefix is a mask of length  $2^{(32-\text{prefix length})}$  bits for IPv4 and  $2^{(128-\text{prefix length})}$  bits for IPv6. Each bit set to 1 represents a valid address. The address is the base IPv4 address plus the position of the bit in the mask where the bits are numbered left to right beginning with zero. For example, the IPv4 addresses 127.2.1.0, 127.2.1.5-127.2.1.15, and 127.2.1.20-127.2.1.29 would be encoded as follows:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+
|1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Those same addresses embedded in IPv6 would be encoded as follows:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+
|1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Type 9 allows a more dense encoding of labels. The label prefix is formatted as a base label value with the non-prefix low-order bits set to zero. The maximum prefix (including leading zeros due to encoding) length is 27. Following the prefix is a mask of length  $2^{(32-\text{prefix length})}$  bits. Each bit set to one represents a valid label. The label is the base label plus the position of the bit in the mask where the bits are numbered left to right beginning with

zero. Label values of all the odd numbers between 1152 and 1279 would be encoded as follows:

```

      0              1              2              3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 0 0 1 0 0 0 0 0 0|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-----+-----+-----+-----+-----+-----+-----+-----+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-----+-----+-----+-----+-----+-----+-----+-----+

```

If the received Multipath Information is non-null, the labels and IP addresses MUST be picked from the set provided. If none of these labels or addresses map to a particular downstream interface, then for that interface, the type MUST be set to 0. If the received Multipath Information is null (i.e., Multipath Length = 0, or for Types 8 and 9, a mask of all zeros), the type MUST be set to 0.

For example, suppose LSR X at hop 10 has two downstream LSRs, Y and Z, for the FEC in question. The received X could return Multipath Type 4, with low/high IP addresses of 127.1.1.1->127.1.1.255 for downstream LSR Y and 127.2.1.1->127.2.1.255 for downstream LSR Z. The head end reflects this information to LSR Y. Y, which has three downstream LSRs, U, V, and W, computes that 127.1.1.1->127.1.1.127 would go to U and 127.1.1.128-> 127.1.1.255 would go to V. Y would then respond with 3 Downstream Mappings: to U, with Multipath Type 4 (127.1.1.1->127.1.1.127); to V, with Multipath Type 4 (127.1.1.127->127.1.1.255); and to W, with Multipath Type 0.

Note that computing Multipath Information may impose a significant processing burden on the receiver. A receiver MAY thus choose to process a subset of the received prefixes. The sender, on receiving a reply to a Downstream Mapping with partial information, SHOULD assume that the prefixes missing in the reply were skipped by the receiver, and MAY re-request information about them in a new echo request.

### 3.3.2. Downstream Router and Interface

The notion of "downstream router" and "downstream interface" should be explained. Consider an LSR X. If a packet that was originated with TTL  $n > 1$  arrived with outermost label L and TTL=1 at LSR X, X

must be able to compute which LSRs could receive the packet if it was originated with TTL=n+1, over which interface the request would arrive and what label stack those LSRs would see. (It is outside the scope of this document to specify how this computation is done.) The set of these LSRs/interfaces consists of the downstream routers/interfaces (and their corresponding labels) for X with respect to L. Each pair of downstream router and interface requires a separate Downstream Mapping to be added to the reply.

The case where X is the LSR originating the echo request is a special case. X needs to figure out what LSRs would receive the MPLS echo request for a given FEC Stack that X originates with TTL=1.

The set of downstream routers at X may be alternative paths (see the discussion below on ECMP) or simultaneous paths (e.g., for MPLS multicast). In the former case, the Multipath Information is used as a hint to the sender as to how it may influence the choice of these alternatives.

#### 3.4. Pad TLV

The value part of the Pad TLV contains a variable number ( $\geq 1$ ) of octets. The first octet takes values from the following table; all the other octets (if any) are ignored. The receiver SHOULD verify that the TLV is received in its entirety, but otherwise ignores the contents of this TLV, apart from the first octet.

Value	Meaning
-----	-----
1	Drop Pad TLV from reply
2	Copy Pad TLV to reply
3-255	Reserved for future use

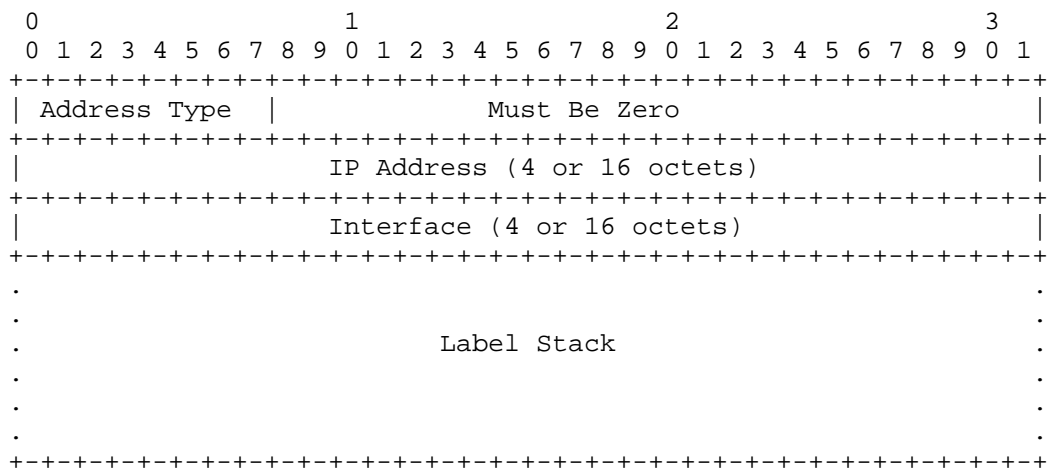
#### 3.5. Vendor Enterprise Number

SMI Private Enterprise Numbers are maintained by IANA. The Length is always 4; the value is the SMI Private Enterprise code, in network octet order, of the vendor with a Vendor Private extension to any of the fields in the fixed part of the message, in which case this TLV MUST be present. If none of the fields in the fixed part of the message have Vendor Private extensions, inclusion of this TLV is OPTIONAL. Vendor Private ranges for Message Types, Reply Modes, and Return Codes have been defined. When any of these are used, the Vendor Enterprise Number TLV MUST be included in the message.

### 3.6. Interface and Label Stack

The Interface and Label Stack TLV MAY be included in a reply message to report the interface on which the request message was received and the label stack that was on the packet when it was received. Only one such object may appear. The purpose of the object is to allow the upstream router to obtain the exact interface and label stack information as it appears at the replying LSR.

The Length is  $K + 4*N$  octets;  $N$  is the number of labels in the label stack. Values for  $K$  are found in the description of Address Type below. The Value field of a Downstream Mapping has the following format:



#### Address Type

The Address Type indicates if the interface is numbered or unnumbered. It also determines the length of the IP Address and Interface fields. The resulting total for the initial part of the TLV is listed in the table below as "K Octets". The Address Type is set to one of the following values:

Type #	Address Type	K Octets
-----	-----	-----
1	IPv4 Numbered	12
2	IPv4 Unnumbered	12
3	IPv6 Numbered	36
4	IPv6 Unnumbered	24

#### IP Address and Interface

IPv4 addresses and interface indices are encoded in 4 octets; IPv6 addresses are encoded in 16 octets.

If the interface upon which the echo request message was received is numbered, then the Address Type MUST be set to IPv4 or IPv6, the IP Address MUST be set to either the LSR's Router ID or the interface address, and the Interface MUST be set to the interface address.

If the interface is unnumbered, the Address Type MUST be either IPv4 Unnumbered or IPv6 Unnumbered, the IP Address MUST be the LSR's Router ID, and the Interface MUST be set to the index assigned to the interface.

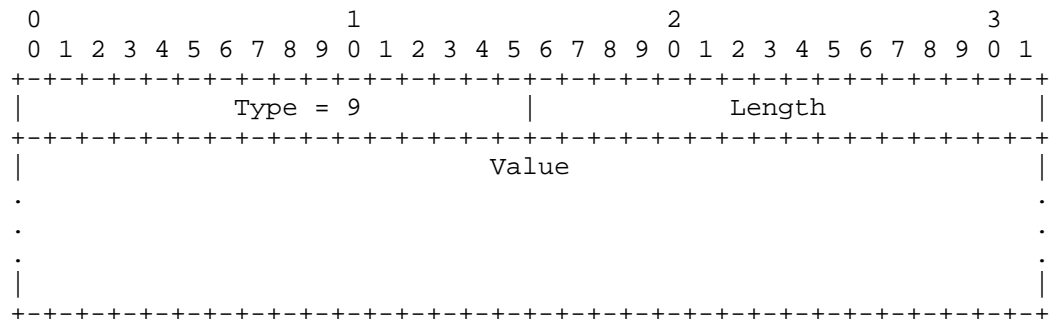
#### Label Stack

The label stack of the received echo request message. If any TTL values have been changed by this router, they SHOULD be restored.

### 3.7. Errored TLVs

The following TLV is a TLV that MAY be included in an echo reply to inform the sender of an echo request of mandatory TLVs either not supported by an implementation or parsed and found to be in error.

The Value field contains the TLVs that were not understood, encoded as sub-TLVs.



### 3.8. Reply TOS Byte TLV

This TLV MAY be used by the originator of the echo request to request that an echo reply be sent with the IP header TOS byte set to the value specified in the TLV. This TLV has a length of 4 with the following value field.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Reply-TOS Byte|                               Must Be Zero          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

#### 4. Theory of Operation

An MPLS echo request is used to test a particular LSP. The LSP to be tested is identified by the "FEC Stack"; for example, if the LSP was set up via LDP, and is to an egress IP address of 10.1.1.1, the FEC Stack contains a single element, namely, an LDP IPv4 prefix sub-TLV with value 10.1.1.1/32. If the LSP being tested is an RSVP LSP, the FEC Stack consists of a single element that captures the RSVP Session and Sender Template that uniquely identifies the LSP.

FEC Stacks can be more complex. For example, one may wish to test a VPN IPv4 prefix of 10.1/8 that is tunneled over an LDP LSP with egress 10.10.1.1. The FEC Stack would then contain two sub-TLVs, the bottom being a VPN IPv4 prefix, and the top being an LDP IPv4 prefix. If the underlying (LDP) tunnel were not known, or was considered irrelevant, the FEC Stack could be a single element with just the VPN IPv4 sub-TLV.

When an MPLS echo request is received, the receiver is expected to verify that the control plane and data plane are both healthy (for the FEC Stack being pinged) and that the two planes are in sync. The procedures for this are in section 4.4 below.

##### 4.1. Dealing with Equal-Cost Multi-Path (ECMP)

LSPs need not be simple point-to-point tunnels. Frequently, a single LSP may originate at several ingresses, and terminate at several egresses; this is very common with LDP LSPs. LSPs for a given FEC may also have multiple "next hops" at transit LSRs. At an ingress, there may also be several different LSPs to choose from to get to the desired endpoint. Finally, LSPs may have backup paths, detour paths, and other alternative paths to take should the primary LSP go down.

To deal with the last two first: it is assumed that the LSR sourcing MPLS echo requests can force the echo request into any desired LSP, so choosing among multiple LSPs at the ingress is not an issue. The problem of probing the various flavors of backup paths that will typically not be used for forwarding data unless the primary LSP is down will not be addressed here.

Since the actual LSP and path that a given packet may take may not be known a priori, it is useful if MPLS echo requests can exercise all



possible paths. This, although desirable, may not be practical, because the algorithms that a given LSR uses to distribute packets over alternative paths may be proprietary.

To achieve some degree of coverage of alternate paths, there is a certain latitude in choosing the destination IP address and source UDP port for an MPLS echo request. This is clearly not sufficient; in the case of traceroute, more latitude is offered by means of the Multipath Information of the Downstream Mapping TLV. This is used as follows. An ingress LSR periodically sends an MPLS traceroute message to determine whether there are multipaths for a given LSP. If so, each hop will provide some information how each of its downstream paths can be exercised. The ingress can then send MPLS echo requests that exercise these paths. If several transit LSRs have ECMP, the ingress may attempt to compose these to exercise all possible paths. However, full coverage may not be possible.

#### 4.2. Testing LSPs That Are Used to Carry MPLS Payloads

To detect certain LSP breakages, it may be necessary to encapsulate an MPLS echo request packet with at least one additional label when testing LSPs that are used to carry MPLS payloads (such as LSPs used to carry L2VPN and L3VPN traffic. For example, when testing LDP or RSVP-TE LSPs, just sending an MPLS echo request packet may not detect instances where the router immediately upstream of the destination of the LSP ping may forward the MPLS echo request successfully over an interface not configured to carry MPLS payloads because of the use of penultimate hop popping. Since the receiving router has no means to differentiate whether the IP packet was sent unlabeled or implicitly labeled, the addition of labels shimmed above the MPLS echo request (using the Nil FEC) will prevent a router from forwarding such a packet out unlabeled interfaces.

#### 4.3. Sending an MPLS Echo Request

An MPLS echo request is a UDP packet. The IP header is set as follows: the source IP address is a routable address of the sender; the destination IP address is a (randomly chosen) IPv4 address from the range 127/8 or IPv6 address from the range 0:0:0:0:0:FFFF:7F00/104. The IP TTL is set to 1. The source UDP port is chosen by the sender; the destination UDP port is set to 3503 (assigned by IANA for MPLS echo requests). The Router Alert IP option of value 0x0 [RFC2113] for IPv4 or value 69 [RFC7506] for IPv6 MUST be set in IP header.

An MPLS echo request is sent with a label stack corresponding to the FEC Stack being tested. Note that further labels could be applied if, for example, the normal route to the topmost FEC in the stack is

via a Traffic Engineered Tunnel [RFC3209]. If all of the FECs in the stack correspond to Implicit Null labels, the MPLS echo request is considered unlabeled even if further labels will be applied in sending the packet.

If the echo request is labeled, one MAY (depending on what is being pinged) set the TTL of the innermost label to 1, to prevent the ping request going farther than it should. Examples of where this SHOULD be done include pinging a VPN IPv4 or IPv6 prefix, an L2 VPN endpoint or a pseudowire. Preventing the ping request from going too far can also be accomplished by inserting a Router Alert label above this label; however, this may lead to the undesired side effect that MPLS echo requests take a different data path than actual data. For more information on how these mechanisms can be used for pseudowire connectivity verification, see [RFC5085].

In "ping" mode (end-to-end connectivity check), the TTL in the outermost label is set to 255. In "traceroute" mode (fault isolation mode), the TTL is set successively to 1, 2, and so on.

The sender chooses a Sender's Handle and a Sequence Number. When sending subsequent MPLS echo requests, the sender SHOULD increment the Sequence Number by 1. However, a sender MAY choose to send a group of echo requests with the same Sequence Number to improve the chance of arrival of at least one packet with that Sequence Number.

The TimeStamp Sent is set to the time-of-day in NTP format that the echo request is sent. The TimeStamp Received is set to zero.

An MPLS echo request MUST have an FEC Stack TLV. Also, the Reply Mode must be set to the desired reply mode; the Return Code and Subcode are set to zero. In the "traceroute" mode, the echo request SHOULD include a Downstream Mapping TLV.

#### 4.4. Receiving an MPLS Echo Request

Sending an MPLS echo request to the control plane is triggered by one of the following packet processing exceptions: Router Alert option, IP TTL expiration, MPLS TTL expiration, MPLS Router Alert label, or the destination address in the 127/8 address range. The control plane further identifies it by UDP destination port 3503.

For reporting purposes the bottom of stack is considered to be stack-depth of 1. This is to establish an absolute reference for the case where the actual stack may have more labels than there are FECs in the Target FEC Stack.

Furthermore, in all the error codes listed in this document, a stack-depth of 0 means "no value specified". This allows compatibility with existing implementations that do not use the Return Subcode field.

An LSR X that receives an MPLS echo request then processes it as follows.

1. General packet sanity is verified. If the packet is not well-formed, LSR X SHOULD send an MPLS Echo Reply with the Return Code set to "Malformed echo request received" and the Subcode to zero. If there are any TLVs not marked as "Ignore" that LSR X does not understand, LSR X SHOULD send an MPLS "TLV not understood" (as appropriate), and the Subcode set to zero. In the latter case, the misunderstood TLVs (only) are included as sub-TLVs in an Errored TLVs TLV in the reply. The header fields Sender's Handle, Sequence Number, and Timestamp Sent are not examined, but are included in the MPLS echo reply message.

The algorithm uses the following variables and identifiers:

Interface-I:	the interface on which the MPLS echo request was received.
Stack-R:	the label stack on the packet as it was received.
Stack-D:	the label stack carried in the Downstream Mapping TLV (not always present)
Label-L:	the label from the actual stack currently being examined. Requires no initialization.
Label-stack-depth:	the depth of label being verified. Initialized to the number of labels in the received label stack S.
FEC-stack-depth:	depth of the FEC in the Target FEC Stack that should be used to verify the current actual label. Requires no initialization.
Best-return-code:	contains the return code for the echo reply packet as currently best known. As the algorithm progresses, this code may change depending on the results of further checks that it performs.
Best-rtn-subcode:	similar to Best-return-code, but for the Echo Reply Subcode.

FEC-status:                result value returned by the FEC Checking algorithm described in section 4.4.1.

/\* Save receive context information \*/

2. If the echo request is good, LSR X stores the interface over which the echo was received in Interface-I, and the label stack with which it came in Stack-R.

/\* The rest of the algorithm iterates over the labels in Stack-R, verifies validity of label values, reports associated label switching operations (for traceroute), verifies correspondence between the Stack-R and the Target FEC Stack description in the body of the echo request, and reports any errors. \*/

/\* The algorithm iterates as follows. \*/

3. Label Validation:

    If Label-stack-depth is 0 {

        /\* The LSR needs to report its being a tail-end for the LSP \*/

        Set FEC-stack-depth to 1, set Label-L to 3 (Implicit Null).  
        Set Best-return-code to 3 ("Replying router is an egress for the FEC at stack depth"), set Best-rtn-subcode to the value of FEC-stack-depth (1) and go to step 5 (Egress Processing).

    }

    /\* This step assumes there is always an entry for well-known label values \*/

    Set Label-L to the value extracted from Stack-R at depth Label-stack-depth. Look up Label-L in the Incoming Label Map (ILM) to determine if the label has been allocated and an operation is associated with it.

    If there is no entry for L {

        /\* Indicates a temporary or permanent label synchronization problem the LSR needs to report an error \*/

        Set Best-return-code to 11 ("No label entry at stack-depth") and Best-rtn-subcode to Label-stack-depth. Go to step 7 (Send Reply Packet).

    }

```
Else {  
    Retrieve the associated label operation from the corresponding  
    NHLFE and proceed to step 4 (Label Operation check).  
}  
  
4. Label Operation Check  
  
If the label operation is "Pop and Continue Processing" {  
    /* Includes Explicit Null and Router Alert label cases */  
  
    Iterate to the next label by decrementing Label-stack-depth and  
    loop back to step 3 (Label Validation).  
}  
  
If the label operation is "Swap or Pop and Switch based on Popped  
Label" {  
  
    Set Best-return-code to 8 ("Label switched at stack-depth") and  
    Best-rtn-subcode to Label-stack-depth to report transit  
    switching.  
  
    If a Downstream Mapping TLV is present in the received echo  
    request {  
  
        If the IP address in the TLV is 127.0.0.1 or 0::1 {  
  
            Set Best-return-code to 6 ("Upstream Interface Index  
            Unknown"). An Interface and Label Stack TLV SHOULD be  
            included in the reply and filled with Interface-I and  
            Stack-R.  
  
        }  
  
        Else {  
  
            Verify that the IP address, interface address, and label  
            stack in the Downstream Mapping TLV match Interface-I and  
            Stack-R. If there is a mismatch, set Best-return-code to  
            5, "Downstream Mapping Mismatch". An Interface and Label  
            Stack TLV SHOULD be included in the reply and filled in  
            based on Interface-I and Stack-R. Go to step 7 (Send  
            Reply Packet).  
  
        }  
  
    }  
}
```

```
}

For each available downstream ECMP path {

    Retrieve output interface from the NHLFE entry.

    /* Note: this return code is set even if Label-stack-depth
    is one */

    If the output interface is not MPLS enabled {

        Set Best-return-code to Return Code 9, "Label switched
        but no MPLS forwarding at stack-depth" and set Best-rtn-
        subcode to Label-stack-depth and goto Send_Reply_Packet.

    }

    If a Downstream Mapping TLV is present {

        A Downstream Mapping TLV SHOULD be included in the echo
        reply (see section 3.3) filled in with information about
        the current ECMP path.

    }

}

If no Downstream Mapping TLV is present, or the Downstream IP
Address is set to the ALLROUTERS multicast address, go to step
7 (Send Reply Packet).

If the "Validate FEC Stack" flag is not set and the LSR is not
configured to perform FEC checking by default, go to step 7
(Send Reply Packet).

/* Validate the Target FEC Stack in the received echo request.

First determine FEC-stack-depth from the Downstream Mapping
TLV. This is done by walking through Stack-D (the Downstream
labels) from the bottom, decrementing the number of labels for
each non-Implicit Null label, while incrementing FEC-stack-
depth for each label. If the Downstream Mapping TLV contains
one or more Implicit Null labels, FEC-stack-depth may be
greater than Label-stack-depth. To be consistent with the
above stack-depths, the bottom is considered to be entry 1.
*/

Set FEC-stack-depth to 0. Set i to Label-stack-depth.
```

```
While (i > 0 ) do {  
    ++FEC-stack-depth.  
    if Stack-D[FEC-stack-depth] != 3 (Implicit Null)  
        --i.  
}  
  
If the number of FECs in the FEC stack is greater than or equal  
to FEC-stack-depth {  
    Perform the FEC Checking procedure (see subsection 4.4.1  
    below).  
  
    If FEC-status is 2, set Best-return-code to 10 ("Mapping for  
    this FEC is not the given label at stack-depth").  
  
    If the return code is 1, set Best-return-code to FEC-return-  
    code and Best-rtn-subcode to FEC-stack-depth.  
}  
  
Go to step 7 (Send Reply Packet).  
}
```

#### 5. Egress Processing:

```
/* These steps are performed by the LSR that identified itself as  
the tail-end LSR for an LSP. */  
  
If received echo request contains no Downstream Mapping TLV, or  
the Downstream IP Address is set to 127.0.0.1 or 0::1 go to step 6  
(Egress FEC Validation).  
  
Verify that the IP address, interface address, and label stack in  
the Downstream Mapping TLV match Interface-I and Stack-R. If not,  
set Best-return-code to 5, "Downstream Mapping Mis-match". A  
Received Interface and Label Stack TLV SHOULD be created for the  
echo response packet. Go to step 7 (Send Reply Packet).
```

#### 6. Egress FEC Validation:

```
/* This is a loop for all entries in the Target FEC Stack starting  
with FEC-stack-depth. */  
  
Perform FEC checking by following the algorithm described in  
subsection 4.4.1 for Label-L and the FEC at FEC-stack-depth.  
  
Set Best-return-code to FEC-code and Best-rtn-subcode to the value  
in FEC-stack-depth.
```

```
If FEC-status (the result of the check) is 1,  
go to step 7 (Send Reply Packet).
```

```
/* Iterate to the next FEC entry */
```

```
++FEC-stack-depth.  
If FEC-stack-depth > the number of FECs in the FEC-stack,  
go to step 7 (Send Reply Packet).
```

```
If FEC-status is 0 {  
  
    ++Label-stack-depth.  
    If Label-stack-depth > the number of labels in Stack-R,  
    Go to step 7 (Send Reply Packet).  
  
    Label-L = extracted label from Stack-R at depth  
    Label-stack-depth.  
    Loop back to step 6 (Egress FEC Validation).  
}
```

#### 7. Send Reply Packet:

Send an MPLS echo reply with a Return Code of Best-return-code, and a Return Subcode of Best-rtn-subcode. Include any TLVs created during the above process. The procedures for sending the echo reply are found in subsection 4.5.

#### 4.4.1. FEC Validation

```
/* This subsection describes validation of an FEC entry within the  
Target FEC Stack and accepts an FEC, Label-L, and Interface-I. The  
algorithm performs the following steps. */
```

1. Two return values, FEC-status and FEC-return-code, are initialized to 0.
2. If the FEC is the Nil FEC {  
  
    If Label-L is either Explicit\_Null or Router\_Alert, return.  
  
    Else {  
  
        Set FEC-return-code to 10 ("Mapping for this FEC is not the  
        given label at stack-depth").  
        Set FEC-status to 1  
        Return.  
    }  
}



```
}
```

3. Check the FEC label mapping that describes how traffic received on the LSP is further switched or which application it is associated with. If no mapping exists, set FEC-return-code to Return 4, "Replying router has no mapping for the FEC at stack-depth". Set FEC-status to 1. Return.
4. If the label mapping for FEC is Implicit Null, set FEC-status to 2 and proceed to step 5. Otherwise, if the label mapping for FEC is Label-L, proceed to step 5. Otherwise, set FEC-return-code to 10 ("Mapping for this FEC is not the given label at stack-depth"), set FEC-status to 1, and return.
5. This is a protocol check. Check what protocol would be used to advertise FEC. If it can be determined that no protocol associated with Interface-I would have advertised an FEC of that FEC-Type, set FEC-return-code to 12 ("Protocol not associated with interface at FEC stack-depth"). Set FEC-status to 1.
6. Return.

#### 4.5. Sending an MPLS Echo Reply

An MPLS echo reply is a UDP packet. It MUST ONLY be sent in response to an MPLS echo request. The source IP address is a routable address of the replier; the source port is the well-known UDP port for LSP ping. The destination IP address and UDP port are copied from the source IP address and UDP port of the echo request. The IP TTL is set to 255. If the Reply Mode in the echo request is "Reply via an IPv4 UDP packet with Router Alert", then the IP header MUST contain the Router Alert IP option of value 0x0 [RFC2113] for IPv4 or 69 [RFC7506] for IPv6. If the reply is sent over an LSP, the topmost label MUST in this case be the Router Alert label (1) (see [RFC3032]).

The format of the echo reply is the same as the echo request. The Sender's Handle, the Sequence Number, and TimeStamp Sent are copied from the echo request; the TimeStamp Received is set to the time-of-day that the echo request is received (note that this information is most useful if the time-of-day clocks on the requester and the replier are synchronized). The FEC Stack TLV from the echo request MAY be copied to the reply.

The replier MUST fill in the Return Code and Subcode, as determined in the previous subsection.

If the echo request contains a Pad TLV, the replier **MUST** interpret the first octet for instructions regarding how to reply.

If the replying router is the destination of the FEC, then Downstream Mapping TLVs **SHOULD NOT** be included in the echo reply.

If the echo request contains a Downstream Mapping TLV, and the replying router is not the destination of the FEC, the replier **SHOULD** compute its downstream routers and corresponding labels for the incoming label, and add Downstream Mapping TLVs for each one to the echo reply it sends back.

If the Downstream Mapping TLV contains Multipath Information requiring more processing than the receiving router is willing to perform, the responding router **MAY** choose to respond with only a subset of multipaths contained in the echo request Downstream Mapping. (Note: The originator of the echo request **MAY** send another echo request with the Multipath Information that was not included in the reply.)

Except in the case of Reply Mode 4, "Reply via application level control channel", echo replies are always sent in the context of the IP/MPLS network.

#### 4.6. Receiving an MPLS Echo Reply

An LSR X should only receive an MPLS echo reply in response to an MPLS echo request that it sent. Thus, on receipt of an MPLS echo reply, X should parse the packet to ensure that it is well-formed, then attempt to match up the echo reply with an echo request that it had previously sent, using the destination UDP port and the Sender's Handle. If no match is found, then X jettisons the echo reply; otherwise, it checks the Sequence Number to see if it matches.

If the echo reply contains Downstream Mappings, and X wishes to traceroute further, it **SHOULD** copy the Downstream Mapping(s) into its next echo request(s) (with TTL incremented by one).

#### 4.7. Issue with VPN IPv4 and IPv6 Prefixes

Typically, an LSP ping for a VPN IPv4 prefix or VPN IPv6 prefix is sent with a label stack of depth greater than 1, with the innermost label having a TTL of 1. This is to terminate the ping at the egress PE, before it gets sent to the customer device. However, under certain circumstances, the label stack can shrink to a single label before the ping hits the egress PE; this will result in the ping terminating prematurely. One such scenario is a multi-AS Carrier's Carrier VPN.

To get around this problem, one approach is for the LSR that receives such a ping to realize that the ping terminated prematurely, and send back error code 13. In that case, the initiating LSR can retry the ping after incrementing the TTL on the VPN label. In this fashion, the ingress LSR will sequentially try TTL values until it finds one that allows the VPN ping to reach the egress PE.

#### 4.8. Non-compliant Routers

If the egress for the FEC Stack being pinged does not support MPLS ping, then no reply will be sent, resulting in possible "false negatives". If in "traceroute" mode, a transit LSR does not support LSP ping, then no reply will be forthcoming from that LSR for some TTL, say,  $n$ . The LSR originating the echo request SHOULD try sending the echo request with  $TTL=n+1$ ,  $n+2$ , ...,  $n+k$  to probe LSRs further down the path. In such a case, the echo request for  $TTL > n$  SHOULD be sent with Downstream Mapping TLV "Downstream IP Address" field set to the ALLROUTERS multicast address until a reply is received with a Downstream Mapping TLV. The label stack MAY be omitted from the Downstream Mapping TLV. Furthermore, the "Validate FEC Stack" flag SHOULD NOT be set until an echo reply packet with a Downstream Mapping TLV is received.

#### 5. Security Considerations

Overall, the security needs for LSP ping are similar to those of ICMP ping.

There are at least three approaches to attacking LSRs using the mechanisms defined here. One is a Denial-of-Service attack, by sending MPLS echo requests/replies to LSRs and thereby increasing their workload. The second is obfuscating the state of the MPLS data plane liveness by spoofing, hijacking, replaying, or otherwise tampering with MPLS echo requests and replies. The third is an unauthorized source using an LSP ping to obtain information about the network.

To avoid potential Denial-of-Service attacks, it is RECOMMENDED that implementations regulate the LSP ping traffic going to the control plane. A rate limiter SHOULD be applied to the well-known UDP port defined below.

Unsophisticated replay and spoofing attacks involving faking or replaying MPLS echo reply messages are unlikely to be effective. These replies would have to match the Sender's Handle and Sequence Number of an outstanding MPLS echo request message. A non-matching replay would be discarded as the sequence has moved on, thus a spoof has only a small window of opportunity. However, to provide a

stronger defense, an implementation MAY also validate the TimeStamp Sent by requiring an exact match on this field.

To protect against unauthorized sources using MPLS echo request messages to obtain network information, it is RECOMMENDED that implementations provide a means of checking the source addresses of MPLS echo request messages against an access list before accepting the message.

It is not clear how to prevent hijacking (non-delivery) of echo requests or replies; however, if these messages are indeed hijacked, LSP ping will report that the data plane is not working as it should.

It does not seem vital (at this point) to secure the data carried in MPLS echo requests and replies, although knowledge of the state of the MPLS data plane may be considered confidential by some. Implementations SHOULD, however, provide a means of filtering the addresses to which echo reply messages may be sent.

Although this document makes special use of 127/8 address, these are used only in conjunction with the UDP port 3503. Furthermore, these packets are only processed by routers. All other hosts MUST treat all packets with a destination address in the range 127/8 in accordance to RFC 1122. Any packet received by a router with a destination address in the range 127/8 without a destination UDP port of 3503 MUST be treated in accordance to RFC 1812. In particular, the default behavior is to treat packets destined to a 127/8 address as "martians".

## 6. IANA Considerations

The TCP and UDP port number 3503 has been allocated by IANA for LSP echo requests and replies.

The following sections detail the new name spaces to be managed by IANA. For each of these name spaces, the space is divided into assignment ranges; the following terms are used in describing the procedures by which IANA allocates values: "Standards Action" (as defined in [RFC5226]), "Specification Required", and "Vendor Private Use".

Values from "Specification Required" ranges MUST be registered with IANA. The request MUST be made via an Experimental RFC that describes the format and procedures for using the code point; the actual assignment is made during the IANA actions for the RFC.

Values from "Vendor Private" ranges MUST NOT be registered with IANA; however, the message MUST contain an enterprise code as registered

with the IANA SMI Private Network Management Private Enterprise Numbers. For each name space that has a Vendor Private range, it must be specified where exactly the SMI Private Enterprise Number resides; see below for examples. In this way, several enterprises (vendors) can use the same code point without fear of collision.

#### 6.1. Message Types, Reply Modes, Return Codes

The IANA has created and will maintain registries for Message Types, Reply Modes, and Return Codes. Each of these can take values in the range 0-255. Assignments in the range 0-191 are via Standards Action; assignments in the range 192-251 are made via "Specification Required"; values in the range 252-255 are for Vendor Private Use, and MUST NOT be allocated.

If any of these fields fall in the Vendor Private range, a top-level Vendor Enterprise Number TLV MUST be present in the message.

Message Types defined in this document are the following:

Value	Meaning
-----	-----
1	MPLS echo request
2	MPLS echo reply

Reply Modes defined in this document are the following:

Value	Meaning
-----	-----
1	Do not reply
2	Reply via an IPv4/IPv6 UDP packet
3	Reply via an IPv4/IPv6 UDP packet with Router Alert
4	Reply via application level control channel

Return Codes defined in this document are listed in section 3.1.

#### 6.2. TLVs

The IANA has created and will maintain a registry for the Type field of top-level TLVs as well as for any associated sub-TLVs. Note the meaning of a sub-TLV is scoped by the TLV. The number spaces for the sub-TLVs of various TLVs are independent.

The valid range for TLVs and sub-TLVs is 0-65535. Assignments in the range 0-16383 and 32768-49161 are made via Standards Action as defined in [RFC5226]; assignments in the range 16384-31743 and 49162-64511 are made via "Specification Required" as defined above;

values in the range 31744-32767 and 64512-65535 are for Vendor Private Use, and MUST NOT be allocated.

If a TLV or sub-TLV has a Type that falls in the range for Vendor Private Use, the Length MUST be at least 4, and the first four octets MUST be that vendor's SMI Private Enterprise Number, in network octet order. The rest of the Value field is private to the vendor.

TLVs and sub-TLVs defined in this document are the following:

Type	Sub-Type	Value Field
----	-----	-----
1		Target FEC Stack
	1	LDP IPv4 prefix
	2	LDP IPv6 prefix
	3	RSVP IPv4 LSP
	4	RSVP IPv6 LSP
	5	Not Assigned
	6	VPN IPv4 prefix
	7	VPN IPv6 prefix
	8	L2 VPN endpoint
	9	"FEC 128" Pseudowire - IPv4 (Deprecated)
	10	"FEC 128" Pseudowire - IPv4
	11	"FEC 129" Pseudowire - IPv4
	12	BGP labeled IPv4 prefix
	13	BGP labeled IPv6 prefix
	14	Generic IPv4 prefix
	15	Generic IPv6 prefix
	16	Nil FEC
	24	"FEC 128" Pseudowire - IPv6
	25	"FEC 129" Pseudowire - IPv6
2		Downstream Mapping
3		Pad
4		Not Assigned
5		Vendor Enterprise Number
6		Not Assigned
7		Interface and Label Stack
8		Not Assigned
9		Errored TLVs
	Any value	The TLV not understood
10		Reply TOS Byte

## 7. Acknowledgements

The original acknowledgements from RFC 4379 state the following:

This document is the outcome of many discussions among many people, including Manoj Leelanivas, Paul Traina, Yakov Rekhter,

Der-Hwa Gan, Brook Bailey, Eric Rosen, Ina Minei, Shivani Aggarwal, and Vanson Lim.

The description of the Multipath Information sub-field of the Downstream Mapping TLV was adapted from text suggested by Curtis Villamizar.

We would like to thank Loa Andersson for motivating the advancement of this bis specification. We also would like to thank Alexander Vainshtein for his review and comments.

## 8. References

### 8.1. Normative References

- [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <<http://www.rfc-editor.org/info/rfc1122>>.
- [RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", RFC 1812, DOI 10.17487/RFC1812, June 1995, <<http://www.rfc-editor.org/info/rfc1812>>.
- [RFC2113] Katz, D., "IP Router Alert Option", RFC 2113, DOI 10.17487/RFC2113, February 1997, <<http://www.rfc-editor.org/info/rfc2113>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <<http://www.rfc-editor.org/info/rfc3032>>.
- [RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, DOI 10.17487/RFC4026, March 2005, <<http://www.rfc-editor.org/info/rfc4026>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<http://www.rfc-editor.org/info/rfc4271>>.

- [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, DOI 10.17487/RFC4379, February 2006, <<http://www.rfc-editor.org/info/rfc4379>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<http://www.rfc-editor.org/info/rfc5905>>.
- [RFC7506] Raza, K., Akiya, N., and C. Pignataro, "IPv6 Router Alert Option for MPLS Operations, Administration, and Maintenance (OAM)", RFC 7506, DOI 10.17487/RFC7506, April 2015, <<http://www.rfc-editor.org/info/rfc7506>>.

## 8.2. Informative References

- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <<http://www.rfc-editor.org/info/rfc792>>.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001, <<http://www.rfc-editor.org/info/rfc3107>>.
- [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, <<http://www.rfc-editor.org/info/rfc3209>>.
- [RFC4365] Rosen, E., "Applicability Statement for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4365, DOI 10.17487/RFC4365, February 2006, <<http://www.rfc-editor.org/info/rfc4365>>.
- [RFC4447] Martini, L., Ed., Rosen, E., El-Aawar, N., Smith, T., and G. Heron, "Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)", RFC 4447, DOI 10.17487/RFC4447, April 2006, <<http://www.rfc-editor.org/info/rfc4447>>.



- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<http://www.rfc-editor.org/info/rfc4761>>.
- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<http://www.rfc-editor.org/info/rfc5036>>.
- [RFC5085] Nadeau, T., Ed. and C. Pignataro, Ed., "Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires", RFC 5085, DOI 10.17487/RFC5085, December 2007, <<http://www.rfc-editor.org/info/rfc5085>>.

## Authors' Addresses

Carlos Pignataro  
Cisco Systems, Inc.

Email: [cpignata@cisco.com](mailto:cpignata@cisco.com)

Nagendra Kumar  
Cisco Systems, Inc.

Email: [naikumar@cisco.com](mailto:naikumar@cisco.com)

Sam Aldrin  
Google

Email: [aldrin.ietf@gmail.com](mailto:aldrin.ietf@gmail.com)

Mach(Guoyi) Chen  
Huawei

Email: [mach.chen@huawei.com](mailto:mach.chen@huawei.com)