TEAS Working Group
Internet Draft
Intended status: Proposed Standard

Vishnu Pavan Beeram (Ed)
Juniper Networks
Ina Minei
Google, Inc
Rob Shakir
British Telecom
Ebben Aries
Facebook
Dante Pacella
Verizon
Tarek Saad
Cisco Systems
Markus Jork
Juniper Networks

Expires: January 3, 2016 July 3, 2015

Implementation Recommendations to Improve the Scalability of RSVP-TE Deployments

draft-beeram-teas-rsvp-te-scaling-rec-00

Status of this Memo

Copyright Notice

Abstract

The scale at which RSVP-TE Label Switched Paths (LSPs) get deployed is growing continually and the onus is on RSVP-TE implementations across the board to keep up with this increasing demand.

This document makes a set of implementation recommendations to help RSVP-TE deployments push the envelope on scaling and advocates the use of a couple of techniques - "Refresh Interval Independent RSVP (RI-RSVP)" and "Per-Peer flow-control" - for improving scaling.

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].

Table of Contents

1. Introduction

The scale at which RSVP-TE [RFC3209] Label Switched Paths (LSPs) get deployed is growing continually and there is considerable onus on RSVP-TE implementations across the board to keep up with this increasing demand in scale.

The set of RSVP Refresh Overhead Reduction procedures [RFC2961] serves as a powerful toolkit for RSVP-TE implementations to help cover a majority of the concerns about soft-state scaling. However, even with these tools in the toolkit, analysis of existing implementations [RFC5439] indicates that the processing required under certain scale may still cause significant disruption to an LSR.

This document builds on the scaling work and analysis that has been done so far and makes a set of concrete implementation recommendations to help RSVP-TE deployments push the envelope further on scaling - push higher the threshold above which an LSR struggles to achieve sufficient processing to maintain LSP state.

This document advocates the use of a couple of techniques - "Refresh Interval Independent RSVP (RI-RSVP)" and "Per-Peer flow-control" - for significantly cutting down the amount of processing cycles required to maintain LSP state. "RI-RSVP" helps completely eliminate RSVP's reliance on refreshes and refresh-timeouts while "Per-Peer Flow-Control" enables a busy RSVP speaker to apply back pressure to its peer(s). In order to reap maximum scaling benefits, it is strongly RECOMMENDED that implementations support both the techniques, but it is possible for an implementation to support just one but not the other.

2. Recommendations

2.1. "RFC2961 specific" Recommendations

The implementation recommendations discussed in this section are based on the proposals made in [RFC2961] and act as pre-requisites for implementing either or both of the techniques discussed in Sections 2.2 and 2.3.

2.1.1. Basic Pre-Requisites

An implementation that supports either or both of the techniques discussed in Sections 2.2 and 2.3:

- SHOULD indicate support for RSVP Refresh Overhead Reduction extensions (as specified in Section 2 of [RFC2961]) by default, with the ability to override the default via configuration.

- MUST support reliable delivery of Path/Resv and the corresponding Tear/Err messages using the procedures specified in [RFC2961].

- MUST support retransmit of all RSVP-TE messages using exponential-backoff, as specified in Section 6 of [RFC2961].

2.1.2. Making Acknowledgements mandatory

The reliable message delivery mechanism specified in [RFC2961] states that "Nodes receiving a non-out of order message containing a MESSAGE_ID object with the ACK_Desired flag set, SHOULD respond with a MESSAGE_ID_ACK object."

In an implementation that supports either or both of the techniques discussed in Sections 2.2 and 2.3, nodes receiving a non-out of order message containing a MESSAGE ID object with the ACK-Desired flag set, MUST respond with a MESSAGE_ID_ACK object. This improvement to the predictability of the system in terms of reliable message delivery is key for being able to take any action based on a non-receipt of an ACK.

2.1.3. Clarifications on reaching Rapid Retry Limit (Rl)

According to section 6 of [RFC2961] "The staged retransmission will continue until either an appropriate MESSAGE_ID_ACK object is received, or the rapid retry limit, Rl, has been reached." The

following clarifies what actions, if any, a router should take once Rl has been reached.

If it is the retransmission of Tear/Err messages and Rl has been reached, the router need not take any further actions. If it is the retransmission of Path/Resv messages and Rl has been reached, then the router starts periodic retransmission of these messages. The retransmitted messages MUST carry MESSAGE_ID object with ACK_Desired flag set. This periodic retransmission SHOULD continue until an appropriate MESSAGE_ID ACK object is received indicating acknowledgement of the (retransmitted) Path/Resv message. The configurable periodic retransmission interval SHOULD be less than the regular refresh interval. A default periodic retransmission interval of 30 seconds is RECOMMENDED by this document.

2.2. Refresh Interval Independent RSVP

The RSVP protocol relies on periodic refreshes for state synchronization between RSVP neighbors and for recovery from lost RSVP messages. It relies on refresh timeout for stale state cleanup. The primary motivation behind introducing the notion of "Refresh Interval Independent RSVP" (RI-RSVP) is to completely eliminate RSVP’s reliance on refreshes and refresh timeouts. This is done by simply increasing the refresh interval to a fairly large value. [RFC2961] and [RFC5439] do talk about increasing the value of the refresh-interval to provide linear improvement on transmission overhead, but also point out the degree of functionality that is lost by doing so. This section revisits this notion, but also proposes sufficient recommendations to make sure that there is no loss of functionality incurred by increasing the value of the refresh interval.

An implementation that supports RI-RSVP:

- MUST support all the recommendations made in Section 2.1

- MUST make the default value of the configurable refresh interval be a large value (10s of minutes). A default value of 20 minutes is RECOMMENDED by this document.

- MUST implement coupling the state of individual LSPs with the state of the corresponding RSVP-TE signaling adjacency. When an RSVP-TE speaker detects RSVP-TE signaling adjacency failure, the speaker MUST act as if the all the Path and Resv state learnt via the failed signaling adjacency has timed out.

- MUST make use of Node-ID based Hello Session ([RFC3209], [RFC4558]) for detection of RSVP-TE signaling adjacency failures; A default value of 9 seconds is RECOMMENDED by this document for the configurable node hello interval (as opposed to the 5ms default value proposed in Section 5.3 of [RFC3209]).

- (If Bypass FRR [RFC4090] is supported,) MUST implement procedures specified in [RI-RSVP-FRR] which describes methods to facilitate FRR that works independently of the refresh-interval.

- MUST indicate support for RI-RSVP via the CAPABILITY object in Hello messages.

2.2.1. Capability Advertisement

An implementation supporting the RI-RSVP recommendations MUST set a new flag "RI-RSVP Capable" in the CAPABILITY object signaled in Hello messages.

The new flag that will be introduced to CAPABILITY object is specified below.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num(134)|  C-Type  (1)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reserved                      |I|T|R|S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

I bit

Indicates that the sender supports RI-RSVP.

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

2.2.2. Compatibility

The RI-RSVP functionality MUST be activated only between peers that indicate their support for this functionality. The RI-RSVP specific Bypass FRR procedures discussed in [RI-RSVP-FRR] introduce a few new

protocol extensions and those MUST get activated only if the participating nodes support RI-RSVP functionality.

2.3. Per-Peer RSVP Flow Control

The set of recommendations discussed in this section provide an RSVP speaker with the ability to apply back pressure to its peer(s) to reduce/eliminate RSVP-TE control plane congestion.

An implementation that supports "Per-Peer RSVP Flow Control":

- MUST support all the recommendations made in Section 2.1

- MUST use lack of ACKs from a peer as an indication of peer’s RSVP-TE control plane congestion. If congestion is detected, the local system MUST throttle RSVP-TE messages to the affected peer. This MUST be done on a per-peer basis. (Per-peer throttling MAY be implemented by a traffic shaping mechanism that proportionally reduces the RSVP signaling packet rate as the number of outstanding Acks increases. And when the number of outstanding Acks decreases, the send rate would be adjusted up again.)

- SHOULD use a Retry Limit (Rl) value of 7 (Section 6.2 of [RFC2961], suggests using 3).

- SHOULD prioritize Tear/Error over trigger Path/Resv (messages that bring up new LSP state) sent to a peer when the local system detects RSVP-TE control plane congestion in the peer.

- MUST indicate support for all recommendations in this section via the CAPABILITY object in Hello messages.

2.3.1. Capability Advertisement

An implementation supporting the "Per-Peer Flow Control" recommendations MUST set a new flag "Per-Peer Flow Control Capable" in the CAPABILITY object signaled in Hello messages.

The new flag that will be introduced to CAPABILITY object is specified below.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num(134)|  C-Type  (1)  |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reserved                    |F|I|T|R|S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

F bit

Indicates that the sender supports Per-Peer RSVP Flow Control

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

2.3.2. Compatibility

The "Per-Peer Flow Control" functionality MUST be activated only if both peers support it. If a peer hasn’t indicated that it is capable of participating in "Per-Peer Flow Control", then it is risky to assume that the peer would always acknowledge a non-out of order message containing a MESSAGE ID object with the ACK-Desired flag set.

2.4. Other Recommendations

The following scaling recommendations have no interdependency with any of the techniques/recommendations specified in Sections 2.2 and 2.3. These are stand-alone functionalities that help improve RSVP-TE scalability.

2.4.1. Summary FRR

If Bypass FRR [RFC4090] is supported by an implementation, it SHOULD support the procedures discussed in [SUMMARY-FRR]. These procedures reduce the amount of RSVP signaling required for Fast Reroute procedures and subsequently improve the scalability of RSVP-TE signaling when undergoing FRR convergence post a link or node failure.

3. Security Considerations

This document does not introduce new security issues. The security considerations pertaining to the original RSVP protocol [RFC2205] and RSVP-TE [RFC3209] and those that are described in [RFC5920] remain relevant.

4. IANA Considerations

4.1. Capability Object Values

[RFC5063] defines the name space for RSVP Capability Object Values. The name space is managed by IANA.

IANA registry: RSVP PARAMETERS

Subsection: Capability Object Values

A Capability flag called "RI-RSVP Capable" is defined in Section 2.2.1 of this document. The bit number for this flag is TBD.

A Capability flag called "Per-Peer Flow Control Capable" is defined in Section 2.3.1 of this document. The bit number for this flag is TBD.

5. References

5.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2205] Braden, R., "Resource Reservation Protocol (RSVP)", RFC 2205, September 1997.

[RFC2961] Berger, L., "RSVP Refresh Overhead Reduction Extensions", RFC 2961, April 2001.

[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.

[RFC4558] Ali, Z., "Node-ID Based Resource Reservation (RSVP) Hello: A Clarification Statement", RFC 4558, June 2006.

[RFC5063] Satyanarayana, A., "Extensions to GMPLS Resource Reservation Protocol Graceful Restart", RFC5063, October 2007.

[RI-RSVP-FRR] Ramachandran, C., "Refresh Interval Independent FRR Facility Protection", draft-chandra-mpls-ri-rsvp-frr,

(work in progress)

[SUMMARY-FRR] Taillon, M., "RSVP-TE Summary Fast Reroute Extensions for LSP Tunnels", draft-mtaillon-mpls-summary-frr-rsvpte-00, (work in progress)

5.2. 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", RFC5920, July 2010.

6. Acknowledgments

The authors would like to thank Yakov Rekhter for initiating this work and providing valuable inputs. They would like to thank Raveendra Torvi and Chandra Ramachandran for participating in the many discussions that led to the recommendations made in this document. They would also like to thank Adrian Farrel for providing detailed review comments.

Appendix A. Recommended Defaults

(a) Refresh-Interval (R)- 20 minutes (Section 2.2)
Given that an implementation supporting RI-RSVP doesn’t rely on refreshes for state sync between peers, the RSVP refresh interval is sort of analogous to IGP refresh interval, the default of which is typically in the order of 10s of minutes. Choosing a default of 20 minutes allows the refresh timer to be randomly set to a value in the range [10 minutes (0.5R), 30 minutes (1.5R)].

(b) Node Hello-Interval - 9 Seconds (Section 2.2)
[RFC3209] defines the hello timeout as 3.5 times the hello interval. Choosing 9 seconds for the node hello-interval gives a hello timeout of 3.5*9 = 31.5 seconds. This puts the hello timeout value to be in the same ballpark as the IGP hello timeout value.

(c) Retry-Limit (Rl) - 7 (Section 2.3)
Choosing 7 as the retry-limit results in an overall rapid retransmit phase of 31.5 seconds. This nicely matches up with the 31.5 seconds hello timeout.

(d) Periodic Retransmission Interval - 30 seconds (Section 2.1.3)

If the Retry-Limit (Rl) is 7, then it takes about 30 (31.5 to be precise) seconds for the 7 rapid retransmit steps to max out. (The last delay from message 6 to message 7 is 16 seconds). The 30 seconds interval also matches the traditional default refresh time.

Authors' Addresses

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

Ina Minei
Google, Inc
Email: inaminei@google.com

Rob Shakir
British Telecom
Email: rob.shakir@bt.com

Ebben Aries
Facebook
Email: exa@fb.com

Dante Pacella
Verizon
Email: dante.j.pacella@verizon.com

Tarek Saad
Cisco Systems
Email: tsaad@cisco.com

Markus Jork
Juniper Networks
Email: mjork@juniper.net

Network Working Group
Internet Draft

Daniele Ceccarelli (Editor)
Ericsson

Intended status: Informational
Expires: September 2015

Young Lee (Editor)
Huawei

March 9, 2015

Framework for Abstraction and Control of Transport Networks

draft-ceccarelli-actn-framework-07.txt

Status of this Memo

Copyright Notice

Abstract

This draft provides a framework for abstraction and control of transport networks.

Table of Contents

1. Introduction

Transport networks have a variety of mechanisms to facilitate separation of data plane and control plane including distributed signaling for path setup and protection, centralized path computation for planning and traffic engineering, and a range of management and provisioning protocols to configure and activate network resources. These mechanisms represent key technologies for enabling flexible and dynamic networking.

Transport networks in this draft refer to a set of different type of connection-oriented networks, primarily Connection-Oriented Circuit Switched (CO-CS) networks and Connection-Oriented Packet Switched (CO-PS) networks. This implies that at least the following transport networks are in scope of the discussion of this draft: Layer 1(L1)

and Layer 0 (L0) optical networks (e.g., Optical Transport Network (OTN), Optical Channel Data Unit (ODU), Optical Channel (OCh)/Wavelength Switched Optical Network (WSON)), Multi-Protocol Label Switching - Transport Profile (MPLS-TP), Multi-Protocol Label Switching - Traffic Engineering (MPLS-TE), as well as other emerging technologies with connection-oriented behavior. One of the characteristics of these network types is the ability of dynamic provisioning and traffic engineering such that resource guarantees can be provided to their clients.

One of the main drivers for Software Defined Networking (SDN) is a decoupling of the network control plane from the data plane. This separation of the control plane from the data plane has been already achieved with the development of MPLS/GMPLS [GMPLS] and PCE [PCE] for TE-based transport networks. One of the advantages of SDN is its logically centralized control regime that allows a global view of the underlying network under its control. Centralized control in SDN helps improve network resources utilization from a distributed network control. For TE-based transport network control, PCE is essentially equivalent to a logically centralized control for path computation function.

Two key aspects that need to be solved by SDN are:

. Network and service abstraction
. End to end coordination of multiple SDN and pre-SDN domains e.g. NMS, MPLS-TE or GMPLS.

As transport networks evolve, the need to provide network and service abstraction has emerged as a key requirement for operators; this implies in effect the virtualization of network resources so that the network is "sliced" for different tenants shown as a dedicated portion of the network resources

Particular attention needs to be paid to the multi-domain case, where Abstraction and Control of Transport Networks (ACTN) can facilitate virtual network operation via the creation of a single virtualized network or a seamless service. This supports operators in viewing and controlling different domains (at any dimension: applied technology, administrative zones, or vendor-specific technology islands) as a single virtualized network.

Network virtualization, in general, refers to allowing the customers to utilize a certain amount of network resources as if they own them and thus control their allocated resources in a way most optimal with higher layer or application processes. This empowerment of customer control facilitates introduction of new services and

applications as the customers are permitted to create, modify, and delete their virtual network services. More flexible, dynamic customer control capabilities are added to the traditional VPN along with a customer specific virtual network view. Customers control a view of virtual network resources, specifically allocated to each one of them. This view is called an abstracted network topology. Such a view may be specific to the set of consumed services as well as to a particular customer. As the Customer Network Controller is envisioned to support a plethora of distinct applications, there would be another level of virtualization from the customer to individual applications.

The framework described in this draft is named Abstraction and Control of Transport Network (ACTN) and facilitates:

- Abstraction of the underlying network resources to higher-layer applications and users (customers); abstraction for a specific application or customer is referred to as virtualization in the ONF SDN architecture. [ONF-ARCH]

- Slicing infrastructure to connect multiple customers to meet specific customer’s service requirements;

- Creation of a virtualized environment allowing operators to view and control multi-subnet multi-technology networks into a single virtualized network;

- Possibility of providing a customer with abstracted network or abstracted services (totally hiding the network).

- A virtualization/mapping network function that adapts customer requests to the virtual resources (allocated to them) to the supporting physical network control and performs the necessary mapping, translation, isolation and security/policy enforcement, etc.; This function is often referred to as orchestration.

- The multi-domain coordination of the underlying transport domains, presenting it as an abstracted topology to the customers via open and programmable interfaces. This allows for the recursion of controllers in a customer-provider relationship.

The organization of this draft is as follows. Section 2 provides a discussion for a Business Model, Section 3 ACTN Architecture, Section 4 ACTN Applicability, and Section 5 ACTN Interface requirements.

2. Business Model of ACTN

The traditional Virtual Private Network (VPN) and Overlay Network (ON) models are built on the premise that one single network provider provides all virtual private or overlay networks to its customers. This model is simple to operate but has some disadvantages in accommodating the increasing need for flexible and dynamic network virtualization capabilities.

The ACTN model is built upon entities that reflect the current landscape of network virtualization environments. There are three key entities in the ACTN model [ACTN-PS]:

- Customers
- Service Providers
- Network Providers

2.1. Customers

Within the ACTN framework, different types of customers may be taken into account depending on the type of their resource needs, on their number and type of access. As example, it is possible to group them into two main categories:

Basic Customer: Basic customers include fixed residential users, mobile users and small enterprises. Usually the number of basic customers is high; they require small amounts of resources and are characterized by steady requests (relatively time invariant). A typical request for a basic customer is for a bundle of voice services and internet access. Moreover basic customers do not modify their services themselves; if a service change is needed, it is performed by the provider as proxy and they generally have very few dedicated resources (subscriber drop), with everything else shared on the basis of some SLA, which is usually best-efforts.

Advanced Customer: Advanced customers typically include enterprises, governments and utilities. Such customers can ask for both point to point and multipoint connectivity with high resource demand significantly varying in time and from customer to customer. This is one of the reasons why a bundled services offer is not enough but it is desirable to provide each of them with customized virtual network

services. Advanced customers may own dedicated virtual resources, or share resources, but shared resources are likely to be governed by more complex SLA agreements; moreover they may have the ability to modify their service parameters directly (within the scope of their virtualized environments. As customers are geographically spread over multiple network provider domains, the necessary control and data interfaces to support such customer needs is no longer a single interface between the customer and one single network provider. With this premise, customers have to interface multiple providers to get their end-to-end network connectivity service and the associated topology information. Customers may have to support multiple virtual network services with different service objectives and QoS requirements. For flexible and dynamic applications, customers may want to control their allocated virtual network resources in a dynamic fashion. To allow that, customers should be given an abstracted view of topology on which they can perform the necessary control decisions and take the corresponding actions. ACTN's primary focus is Advanced Customers.

Customers of a given service provider can in turn offer a service to other customers in a recursive way. An example of recursiveness with 2 service providers is shown below.

- Customer (of service B)
- Customer (of service A) & Service Provider (of service B)
- Service Provider (of service A)
- Network Provider

```
+-------------------------------------------------------------+   ---
|                                                             |    ^
|                                     Customer (of service B)|    .
| +-----------------------------------------------------------+ |    B
| |                                                           | |--- .
| |Customer (of service A) & Service Provider(of service B)| | ^  .
| | +-------------------------------------------------+  | | .  .
| | |                                                 |  | | .  .
| | |                     Service Provider (of service A)|  | | A  .
| | |+-----------------------------------------+       |  | | .  .
| | ||                                         |       |  | | .  .
| | ||                          Network provider|       |  | | v  v
| | |+-----------------------------------------+       |  | |------
| | +-------------------------------------------------+  | |
| +-----------------------------------------------------------+ |
+-------------------------------------------------------------+
```

Figure 1: Network Recursiveness.

2.2. Service Providers

Service providers are the providers of virtual network services to their customers. Service providers may or may not own physical network resources. When a service provider is the same as the network provider, this is similar to traditional VPN models. This model works well when the customer maintains a single interface with a single provider. When customer location spans across multiple independent network provider domains, then it becomes hard to facilitate the creation of end-to-end virtual network services with this model.

A more interesting case arises when network providers only provide infrastructure while service providers directly interface their customers. In this case, service providers themselves are customers of the network infrastructure providers. One service provider may need to keep multiple independent network providers as its end-users span geographically across multiple network provider domains.

```
Customer            X -----------------------------------X

Service Provider A  X -----------------------------------X

Network Provider B                     X-----------------X

Network Provider A  X------------------X
```

The ACTN network model is predicated upon this three tier model and is summarized in figure below:

```
                     +----------------------+
                     |       customer       |
                     +----------------------+
                               |
                               |   /\  Service/Customer specific
                               |   ||  Abstract Topology
                               |   ||
                     +----------------------+  E2E abstract
                     |  Service Provider    | topology creation
                     +----------------------+
                     /         |            \
```

```
                     /          |             \  Network Topology
                    /           |              \ (raw or abstract)
                   /            |               \
   +------------------+   +------------------+   +------------------+
   |Network Provider 1|   |Network Provider 2|   |Network Provider 3|
   +------------------+   +------------------+   +------------------+
```

Figure 2: Three tier model.

There can be multiple types of service providers.

- Data Center providers: can be viewed as a service provider type as they own and operate data center resources to various WAN clients, they can lease physical network resources from network providers.
- Internet Service Providers (ISP): can be a service provider of internet services to their customers while leasing physical network resources from network providers.
- Mobile Virtual Network Operators (MVNO): provide mobile services to their end-users without owning the physical network infrastructure.

The network provider space is the one where recursiveness occurs. A customer-provider relationship between multiple service providers can be established leading to a hierarchical architecture of controllers within service provider network.

2.3. Network Providers

Network Providers are the infrastructure providers that own the physical network resources and provide network resources to their customers. The layered model proposed by this draft separates the concerns of network providers and customers, with service providers acting as aggregators of customer requests.

3. ACTN architecture

This section provides a high-level control and interface model of ACTN.

The ACTN architecture, while being aligned with the ONF SDN architecture [ONF-ARCH], is presenting a 3-tiers reference model. It allows for hierarchy and recursiveness not only of SDN controllers but also of traditionally controlled domains. It defines three types of controllers depending on the functionalities they implement. The main functionalities that are identified are:

. Multi domain coordination function: With the definition of domain being "everything that is under the control of the same controller",it is needed to have a control entity that oversees the specific aspects of the different domains and to build a single abstracted end-to-end network topology in order to coordinate end-to-end path computation and path/service provisioning.

. Virtualization/Abstraction function: To provide an abstracted view of the underlying network resources towards customer, being it the client or a higher level controller entity. It includes computation of customer resource requests into virtual network paths based on the global network-wide abstracted topology and the creation of an abstracted view of network slices allocated to each customer, according to customer-specific virtual network objective functions, and to the customer traffic profile.

. Customer mapping function: In charge of mapping customer VN setup commands into network provisioning requests to the Physical Network Controller (PNC) according to business OSS/NMS provisioned static or dynamic policy. Moreover it provides mapping and translation of customer virtual network slices into physical network resources

. Virtual service coordination: Virtual service coordination function in ACTN incorporates customer service-related knowledge into the virtual network operations in order to seamlessly operate virtual networks while meeting customer’s service requirements.

The functionality is covering two types of services:

- Service-aware Connectivity Services: This category includes all the network service operations used to provide connectivity between customer end-points while meeting policies and service related constraints. The data model for this category would include topology entities such as

virtual nodes, virtual links, adaptation and termination points and service-related entities such as policies and service related constraints. (See Section 4.2.2)

- Network Function Virtualization Services: These kinds of services are usually setup between customers' premises and service provider premises and are provided mostly by cloud providers or content delivery providers. The context may include, but not limited to a security function like firewall, a traffic optimizer, the provisioning of storage or computation capacity where the customer does not care whether the service is implemented in a given data center or another. These services may be hosted virtually by the provider or physically part of the network. This allows the service provider to hide his own resources (both network and data centers) and divert customer requests where most suitable. This is also known as "end points mobility" case and introduces new concepts of traffic and service provisioning and resiliency. (e.g. Virtual Machine mobility)." (See Section 4.2.3)

About the Customer service-related knowledge it includes:

- VN Service Requirements: The end customer would have specific service requirements for the VN including the customer endpoints access profile as well as the E2E customer service objectives. The ACTN framework architectural "entities" would monitor the E2E service during the lifetime of VN by focusing on both the connectivity provided by the network as well as the customer service objectives. These E2E service requirements go beyond the VN service requirements and include customer infrastructure as well.

- Application Service Policy: Apart for network connectivity, the customer may also require some policies for application specific features or services. The ACTN framework would take these application service policies and requirements into consideration while coordinating the virtual network operations, which require end customer connectivity for these advanced services.

While the "types" of controller defined are shown in Figure 3 below and are the following:

. CNC - Customer Network Controller

. MDSC - Multi Domain Service Coordinator
. PNC - Physical Network Controller

```
   VPN customer         NW Mobile Customer     ISP NW service Customer
       |                        |                          |
   +-------+                +-------+                  +-------+
   | CNC-A |                | CNC-B |                  | CNC-C |
   +-------+                +-------+                  +-------+
         \___________           |             ______________/
          ----------            |             ------------
                     \          |            /
                      +-----------------------+
                      |         MDSC          |
                      +-----------------------+
            __________/          |            \_________
            ----------           |             ------------____
           /                     |                         \
   +-------+                +-------+                  +-------+
   |  PNC  |                |  PNC  |                  |  PNC  |
   +-------+                +-------+                  +-------+
        | GMPLS            /      |                    /    \
        | trigger         /       |                   /      \
       --------       __----      +-----+  __        +-----+     \
      (        )       (    )_    | PNC |__          | PCE |      \
      -        -      ( Phys )    +-----+            +-----+    -----
     (  GMPLS   )      (Netw)        |                /        (     )
    (  Physical  )      ----         |               /        ( Phys. )
     (  Network )                   -----        -----          ( Net )
      -        -                   (     )      (     )          -----
      (        )                  ( Phys. )    ( Phys  )
       --------                   ( Net )      ( Net )
                                   -----        -----
```

Figure 3: ACTN Control Hierarchy

3.1. Customer Network Controller

A Virtual Network Service is instantiated by the Customer Network Controller via the CMI (CNC-MDSC Interface). As the Customer Network Controller directly interfaces the application stratum, it understands multiple application requirements and their service needs. It is assumed that the Customer Network Controller and the MDSC have a common knowledge on the end-point interfaces based on their business negotiation prior to service instantiation. End-point interfaces refer to customer-network physical interfaces that connect customer premise equipment to network provider equipment. Figure 10 in Appendix shows an example physical network topology that supports multiple customers. In this example, customer A has

three end-points A.1, A.2 and A.3. The interfaces between customers and transport networks are assumed to be 40G OTU links.

In addition to abstract networks, ACTN allows to provide the CNC with services. Example of services include connectivity between one of the customer’s end points with a given set of resources in a data center from the service provider.

3.2. Multi Domain Service Coordinator

The MDSC (Multi Domain Service Coordinator) sits between the CNC (the one issuing connectivity requests) and the PNCs (Physical Network Controllersr - the ones managing the physical network resources). The MDSC can be collocated with the PNC, especially in those cases where the service provider and the network provider are the same entity.

The internal system architecture and building blocks of the MDSC are out of the scope of ACTN. Some examples can be found in the Application Based Network Operations (ABNO) architecture [ABNO] and the ONF SDN architecture [ONF-ARCH].

The MDSC is the only building block of the architecture that is able to implement all the four ACTN main functionalities, i.e. multi domain coordination function, virtualization/abstraction function, customer mapping function and virtual service coordination.
A hierarchy of MDSCs can be foreseen for scalability and administrative choices.

```
   +-------+                 +-------+                 +-------+
   | CNC-A |                 | CNC-B |                 | CNC-C |
   +-------+                 +-------+                 +-------+
         \___________            |             ___________/
          ----------             |             ----------
                     \           |            /
                      +-----------------------+
                      |         MDSC          |
                      +-----------------------+
            __________/          |            \__________
             ----------          |             -----------____
           /                     |                        \
   +----------+              +----------+             +--------+
   |   MDSC   |              |   MDSC   |             |  MDSC  |
   +----------+              +----------+             +--------+
        |                   /     |                     /    \
        |                  /      |                    /      \
     +-----+           +-----+  +-----+            +-----+  +-----+
     | PNC |           | PNC |  | PNC |            | PNC |  | PNC |
     +-----+           +-----+  +-----+            +-----+  +-----+
```

Figure 4: Controller recursiveness

A key requirement for allowing recursion of MDSCs is that a single interface needs to be defined both for the north and the south bounds.
In order to allow for multi-domain coordination a 1:N relationship must be allowed between MDSCs and between MDSCs and PNCs (i.e. 1 parent MDSC and N child MDSC or 1 MDSC and N PNCs). In addition to that it could be possible to have also a M:1 relationship between MDSC and PNC to allow for network resource partitioning/sharing among different customers not necessarily connected to the same MDSC (e.g. different service providers).
It should be noted that the interface between the parent MDSC and a child MDSC does not introduce any complexity as it is "internal" and "transparent" from the perspective of the CNCs and the PNCs and it makes use of the same interface model and its primitives as the CMI and MPI.

3.3. Physical Network Controller

The physical network controller is the one in charge of configuring the network elements, monitoring the physical topology of the network and passing it, either raw or abstracted, to the MDSC.

The internal architecture of the PNC, his building blocks and the way it controls its domain, are out of the scope of ACTN. Some examples can be found in the Application Based Network Operations (ABNO) architecture [ABNO] and the ONF SDN architecture [ONF-ARCH]

The PNC, in addition to being in charge of controlling the physical network, is able to implement two of the four ACTN main functionalities: multi domain coordination function and virtualization/abstraction function
A hierarchy of PNCs can be foreseen for scalability and administrative choices.

3.4. ACTN interfaces

To allow virtualization and multi domain coordination, the network has to provide open, programmable interfaces, in which customer applications can create, replace and modify virtual network resources and services in an interactive, flexible and dynamic fashion while having no impact on other customers. Direct customer control of transport network elements and virtualized services is not perceived as a viable proposition for transport network providers due to security and policy concerns among other reasons. In addition, as discussed in the previous section, the network control plane for transport networks has been separated from data plane and as such it is not viable for the customer to directly interface with transport network elements.

While the current network control plane is well suited for control of physical network resources via dynamic provisioning, path computation, etc., a multi service domain controller needs to be built on top of physical network controller to support network virtualization. On a high-level, virtual network control refers to a mediation layer that performs several functions:

Figure 4 depicts a high-level control and interface architecture for ACTN. A number of key ACTN interfaces exist for deployment and operation of ACTN-based networks. These are highlighted in Figure 4 (ACTN Interfaces) below:

```
                 .--------------
                -------------   |
               | Application |--
                -------------
                      ^
                      | I/F A                 --------
                      v                      (        )
                 --------------             -          -
                | Customer     |           (  Customer  )
                |  Network     |---------->(    Network   )
                |   Controller |           (            )
                 --------------             -          -
                      ^                      (        )
                      | I/F B                 --------
                      v                       ^    ^
                 --------------               :    :
                | MultiDomain  |               :     .
                |  Service     |               :      .
                |   Coordinator|            --------   . I/F E
                 --------------            (        )   .
                      ^                   -          -   .
                      | I/F C            (  Physical  )   .
                      v                 (    Network   )   .
                   ---------------       (            )     --------
                  |               |<----> -          -     (        )
                 --------------   |        (        )     -         -
                | Physical     |--          --------     (  Physical  )
                |  Network     |<----------------------->(    Network   )
                |   Controller |         I/F D           (            )
                 --------------                          -          -
                                                          (        )
                                                           --------
```

Figure 4: ACTN Interfaces

The interfaces and functions are described below:

. Interface A: A north-bound interface (NBI) that will communicate the service request or application demand. A request will include specific service properties, including: services, topology, bandwidth and constraint information.

. Interface B: The CNC-MDSC Interface (CMI) is an interface between a Customer Network Controller and a Multi Service Domain Controller. It requests the creation of the network resources, topology or services for the applications. The Virtual Network Controller may also report potential network topology availability if queried for current capability from the Customer Network Controller.

. Interface C: The MDSC-PNC Interface (MPI) is an interface between a Multi Domain Service Coordinator and a Physical Network Controller. It communicates the creation request, if required, of new connectivity of bandwidth changes in the physical network, via the PNC. In multi-domain environments, the MDSC needs to establish multiple MPIs, one for each PNC, as there are multiple PNCs responsible for its domain control.

. Interface D: The provisioning interface for creating forwarding state in the physical network, requested via the Physical Network Controller.

. Interface E: A mapping of physical resources to overlay resources.

The interfaces within the ACTN scope are B and C.

3.5. Work in Scope of ACTN

This section provides a summary of use-cases in terms of two categories: (i) service-specific requirements; (ii) network-related requirements.

Service-specific requirements listed below are uniquely applied to the work scope of ACTN. Service-specific requirements are related to virtual service coordination function defined in Section 3. These requirements are related to customer’s VNs in terms of service policy associated with VNs such as service performance objectives, VN endpoint location information for certain required service-specific functions (e.g., security and others), VN survivability requirement, or dynamic service control policy, etc.

Network-related requirements are related to virtual network operation function defined in Section 3. These requirements are related to multi-domain and multi-layer signaling, routing, protection/restoration and synergy, re-optimization/re-grooming, etc. These requirements are not inherently unique for the scope of ACTN but some of these requirements are in scope of ACTN, especially for coherent/seamless operation aspect of multiple controller hierarchy.

The following table gives an overview of service-specific requirements and network-related requirements respectively for each ACTN use-case and identifies the work in scope of ACTN.

Details on these requirements will be developed into the information model in [ACTN-Info].

Use-case	Service-specific Requirements	Network-related Requirements	ACTN Work Scope
[Cheng]	- E2E service provisioning - Performance monitoring - Resource utilization abstraction	- Multi-layer (L2/L2.5) coordination - VNO for multi-domain transport networks	- Dynamic multi-layer coordination based on utilization is in scope of ACTN - YANG for utilization abstraction
[Dhody]	- Service awareness/ coordination between P/O.	- POI Performance monitoring - Protection/ Restoration synergy	- Performance related data model may be in scope of ACTN - Customer’s VN survivability policy enforcement for protection/restoration is unique to ACTN.
[Fang]	- Dynamic VM migration (service), Global load balancing (utilization efficiency), Disaster recovery - Service-aware network	- On-demand virtual circuit request - Network Path Connection request	- Multi-destination service selection policy enforcement and its related primitives/information are unique to

	query - Service Policy Enforcement		ACTN. - Service-aware network query and its data model can be extended by ACTN.
[Klee]		- Two stage path computation E2E signaling coordination - Abstraction of inter-domain info - Enforcement of network policy (peering, domain preference) - Network capability exchange (pull/push, abstraction level, etc.)	- Multi-domain service policy coordination to network primitives is in scope of ACTN
[Kumaki]	- On-demand VN creation - Multi-service level for VN - VN survivability /diversity/confidentiality		- All of the service-specific lists in the left column is unique to ACTN.
[Lopez]	- E2E accounting and resource usage data	- E2E connection management, path provisioning - E2E network	- Escalation of performance and fault management

	- E2E service policy enforcement	monitoring and fault management	data to CNC and the policy enforcement for this area is unique to ACTN.
[Shin]	- Current network resource abstraction Endpoint/DC dynamic selection (for VM migration)	- LB for recovery - Multi-layer routing and optimization coordination	- Multi-layer routing and optimization are related to VN's dynamic endpoint selection policy.
[Xu]	- Dynamic service control policy enforcement - Dynamic service control	- Traffic monitoring - SLA monitoring	- Dynamic service control policy enforcement and its control primitives are in scope of ACTN - Data model to support traffic monitoring data is an extension of YANG model ACTN can extend.

4. References

4.1. Informative References

[PCE] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", IETF RFC 4655, August 2006.

[PCE-S] Crabbe, E, et. al., "PCEP extension for stateful PCE",draft-ietf-pce-stateful-pce, work in progress.

[GMPLS] Manning, E., et al., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, October 2004.

[NFV-AF] "Network Functions Virtualization (NFV); Architectural Framework", ETSI GS NFV 002 v1.1.1, October 2013.

[ACTN-PS] Y. Lee, D. King, M. Boucadair, R. Jing, L. Contreras Murillo, "Problem Statement for Abstraction and Control of Transport Networks", draft-leeking-actn-problem-statement, work in progress.

[ONF] Open Networking Foundation, "OpenFlow Switch Specification Version 1.4.0 (Wire Protocol 0x05)", October 2013.

[ABNO] King, D., and Farrel, A., "A PCE-based Architecture for Application-based Network Operations", draft-farrkingel-pce-abno-architecture, work in progress.

[ACTN-Info] Y. Lee, S. Belotti, D. Dhody, "Information Model for Abstraction and Control of Transport Networks", draft-leebelotti-teas-actn-info, work in progress.

[Cheng] W. Cheng, et. al., "ACTN Use-cases for Packet Transport Networks in Mobile Backhaul Networks", draft-cheng-actn-ptn-requirements, work in progress.

[Dhody] D. Dhody, et. al., "Packet Optical Integration (POI) Use Cases for Abstraction and Control of Transport Networks (ACTN)", draft-dhody-actn-poi-use-case, work in progress.

[Fang] L. Fang, "ACTN Use Case for Multi-domain Data Center Interconnect", draft-fang-actn-multidomain-dci, work in progress.

[Klee] K. Lee, H. Lee, R. Vilata, V. Lopez, "ACTN Use-case for On-demand E2E Connectivity Services in Multiple Vendor Domain Transport Networks", draft-klee-actn-connectivity-multi-vendor-domains, work in progress.

[Kumaki] K. Kumaki, T. Miyasaka, "ACTN : Use case for Multi Tenant VNO ", draft-kumaki-actn-multitenant-vno, work in progress.

[Lopez] D. Lopez (Ed), "ACTN Use-case for Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-lopez-actn-vno-multidomains, work in progress.

[Shin] J. Shin, R. Hwang, J. Lee, "ACTN Use-case for Mobile Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-shin-actn-mvno-multi-domain, work in progress.

[Xu] Y. Xu, et. al., "Use Cases and Requirements of Dynamic Service Control based on Performance Monitoring in ACTN Architecture", draft-xu-actn-perf-dynamic-service-control, work in progress.

5. Contributors

Authors' Addresses

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

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

Luyuan Fang
Email: luyuanf@gmail.com

Diego Lopez
Telefonica I+D
Don Ramon de la Cruz, 82
28006 Madrid, Spain
Email: diego@tid.es

Sergio Belotti
Alcatel Lucent
Via Trento, 30
Vimercate, Italy
Email: sergio.belotti@alcatel-lucent.com

Daniel King
Lancaster University
Email: d.king@lancaster.ac.uk

Dhruv Dhoddy
Huawei Technologies
dhruv.ietf@gmail.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: January 21, 2018

H. Chen
Huawei Technologies
M. Toy
Verizon
V. Liu
China Mobile
L. Liu
Fijitsu

July 20, 2017

Extensions to MPLS for Temporal LSP
draft-chen-teas-rsvp-tts-02.txt

Abstract

This document specifies extensions to RSVP-TE for creating and maintaining a Traffic Engineering (TE) Label Switched Path (LSP) in a time interval or a sequence of time intervals.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

Once an existing multiprotocol label switching (MPLS) traffic engineering (TE) label switched path (LSP) is set up, it is assumed to carry traffic forever until it is down. When an MPLS TE LSP tunnel is up, it is assumed that the LSP consumes its reserved network resources forever even though the LSP may only use network resources during some period of time. As a result, the network resources are not used efficiently. Moreover, a tunnel service can not be reserved or booked in advance in a period of time.

This document specifies extensions to RSVP-TE for creating and maintaining an MPLS TE LSP in a period of time called a time interval or a sequence of time intervals. It is assumed that the LSP carries traffic during this time interval or each of these time intervals. Thus the network resources are efficiently used. More importantly, some new services can be provided. For example, a consumer can book a tunnel service in advance for a given time interval. Tunnel services may be scheduled as requested.

2. Terminology

A Time Interval: a time period from time Ta to time Tb.

LSP: Label Switched Path. An LSP is a P2P (point-to-point) LSP or a P2MP (point-to-multipoiint) LSP.

LSP in a time interval: LSP that carries traffic in the time interval.

LSP in a sequence of time intervals: LSP that carries traffic in each of the time intervals.

Temporal LSP: LSP in a time interval or LSP in a sequence of time intervals.

TEDB: Traffic Engineering Database.

This document uses terminologies defined in RFC3209 and RFC4875.

3. Conventions Used in This Document

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

4. Temporal LSP Overview

This section briefs the architecture for supporting temporal LSPs and some operations on temporal LSPs.

4.1. Architecture Overview

Based on the existing architecture for supporting TE LSPs, we can extend a few of components to support temporal LSPs. These components include OSPF, CSPF and RSVP-TE.

OSPF is extended to distribute and maintain TE inforamtion for a link in a sequence of time intervals. CSPF is extended to compute a path for a temporal LSP based on the TEDB containing TE information for every link in a sequence of time intervals. RSVP-TE is extended to create a temporal LSP and maintain the status of the temporal LSP.

4.2. Operations Overview

On the ingress of a temporal LSP, a user configures it with a time interval or a sequence of time intervals. A simple time interval is a time interval from start time Ta to end time Tb, which may be represented as [Ta, Tb].

When an LSP is configured with time interval [Ta, Tb], a path satisfying the constraints for the LSP in the time interval is computed and the LSP along the path is set up to carry traffic from Ta to Tb.

For time interval from start time Ta to infinite as end time, it may be represented as [Ta, INFINITE].

In addition to simple time intervals, there are recurrent time intervals and elastic time intervals.

A recurrent time interval represents a series of repeated simple time intervals. It has a simple time interval such as [Ta, Tb], a number of repeats such as 10 (repeats 10 times), and a repeat cycle/time such as a week (repeats every week).

Recurrent time interval "[Ta, Tb] repeats n times with repeat cycle C" represents n+1 simple time intervals as follows:

[Ta, Tb], [Ta+C, Tb+C], [Ta+2C, Tb+2C], . . ., [Ta+nC, Tb+nC]

When an LSP is configured with a recurrent time interval such as "[Ta, Tb] repeats 10 times with a repeat cycle a week", a path

satisfying the constraints for the LSP in each of the simple time intervals (such as 11 simple time intervals) represented by the recurrent time interval is computed and the LSP along the path is set up to carry traffic in each of the simple time intervals.

An elastic time interval represents a time period with an elastic range. It has a simple time interval such as [Ta, Tb] with an elastic range such as within -P and Q.

Elastic time interval "[Ta, Tb] within -P and Q" means a time period from (Ta+X) to (Tb+X), where -P <= X <= Q, P and Q is an amount of time such as 600 seconds.

When an LSP is configured with an elastic time interval such as "[Ta, Tb] within -P and Q", a path is computed such that the path satisfies the constraints for the LSP in the time period from (Ta+X) to (Tb+X) and |X| is the minimum value from -P to Q. That is that [Ta+X, Tb+X] is the time interval closest to [Ta, Tb] within the elastic range. The LSP along the path is set up to carry traffic in the time period from (Ta+X) to (Tb+X).

5. TIME INTERVAL Object

This section presents a few of TIME-INTERVAL objects, which are the internal representations of time intervals. A Class-Num for the objects is TBD, which is to be assigned by IANA.

5.1. Absolute TIME INTERVAL Object

The format of an absolute TIME-INTERVAL object body is illustrated below.

```
     Class-Name: TIME-INTERVAL,   Class-Num: TBD,   C-Type = 1
   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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                          Start-time                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                           End-time                            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  o Start-time: The time LSP starts to carry traffic
  o End-time:   The time LSP ends carrying traffic
```

An absolute TIME-INTERVAL object contains a Start-time and an End-time, representing time interval [Start-time, End-time]. All bits in

End-time field set to one represents INFINITE. Both of these two times are the times that are synchronized among all network nodes.

Thus the clocks on all the nodes MUST be synchronized if an absolute TIME-INTERVAL object is used. The time period represented in an absolute TIME-INTERVAL object is more accurate.

5.2. Relative TIME INTERVAL Object

The format of a relative TIME-INTERVAL object body is shown below.

```
     Class-Name: TIME-INTERVAL,   Class-Num: TBD,   C-Type = 2
   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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                       Start-time-length                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        End-time-length                        |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  o Start-time-length: The time length in seconds from current time
                       to the time LSP starts to carry traffic
  o End-time-length:   The time length in seconds from current time
                       to the time LSP ends carrying traffic
```

A relative TIME-INTERVAL object contains a Start-time-length and an End-time-length, which represents time interval below:

```
 [current-time + Start-time-length, current-time + End-time-length]
```

where current-time is the current local time on a node. All bits in End-time-length field set to one represents INFINITE.

When a time interval from time Ta to time Tb is configured on a node, these two time lengths are the time lengths that are computed on the node using the current local time as follows.

```
  Start-time-length = Ta - current-time;
  End-time-length   = Tb - current-time;
```

For a relative TIME-INTERVAL object, the clocks/times on all the nodes can be different.

5.3. Recurrent Absolute TIME INTERVAL Object

For a recurrent absolute TIME-INTERVAL object, its body contains a Start-time, an End-time, a Repeat-time-length, a Options field and a Number-repeats field. The format of its body is illustrated below:

The Start-time and End-time represents time interval [Start-time, End-time]. The Repeat-time-length represents a repeat cycle/time, which is valid if the Options field is set to indicate the way to repeat is "repeat every Repeat-time-length". The Options field indicates a way to repeat. The Number-repeats indicates the number of repeats of time interval [Start-time, End-time].

```
     Class-Name: TIME-INTERVAL,   Class-Num: TBD,   C-Type = 3
   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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                          Start-time                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                           End-time                            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                      Repeat-time-length                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Options |         Number-repeats          |    Reserved(0)    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Start-time: The time LSP starts to carry traffic.

End-time: The time LSP ends carrying traffic.

Repeat-time-length: The time length in seconds after which LSP starts to carry traffic again for (End-time - Start-time).

Options: Indicates a way to repeat.

Options = 1: repeat every day;

Options = 2: repeat every week;

Options = 3: repeat every month;

Options = 4: repeat every year;

Options = 5: repeat every Repeat-time-length.

Number-repeats: The number of repeats. In each of repeats, LSP carries traffic.

5.4. Recurrent Relative TIME INTERVAL Object

For a recurrent relative TIME-INTERVAL object, the format of its body is illustrated below. it contains a Start-time-length, an End-time-length, a Repeat-time-length, a Options field and a Number-repeats field.

The Start-time-length and End-time-length represents time interval

[current-time + Start-time-length, current-time + End-time-length]

where current-time is a current local time.

The Repeat-time-length represents a repeat cycle/time, which is valid if the Options field is set to indicate the way to repeat is "repeat every Repeat-time-length". The Options field indicates a way to repeat. The Number-repeats indicates the number of repeats of the time interval above.

```
     Class-Name: TIME-INTERVAL,   Class-Num: TBD,   C-Type = 4
   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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                       Start-time-length                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        End-time-length                        |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                      Repeat-time-length                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Options |         Number-repeats          |    Reserved (0)   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Start-time-length: The time length in seconds from a current local time to the time LSP starts to carry traffic.

End-time-length: The time length in seconds from a current local time to the time LSP ends carrying traffic.

Repeat-time-length: The time length in seconds after which LSP starts to carry traffic again for (End-time-length - Start-time-length).

Options: Indicates a way to repeat.

Options = 1: repeat every day;

Options = 2: repeat every week;

Options = 3: repeat every month;

Options = 4: repeat every year;

Options = 5: repeat every Repeat-time-length.

Number-repeats: The number of repeats. In each of repeats, LSP carries traffic.

6. Path Message

A Path message is enhanced to carry the information about a time interval or a sequence of time intervals through including a time interval list. The format of the message is illustrated below.

```
<Path Message> ::= <Common Header> [ <INTEGRITY> ]
                   [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
                   [ <MESSAGE_ID> ]<SESSION> <RSVP_HOP> <TIME_VALUES>
                   [ <EXPLICIT_ROUTE> ]
                   <LABEL_REQUEST> [ <PROTECTION> ] [ <LABEL_SET> ...]
                   [ <SESSION_ATTRIBUTE> ] [ <NOTIFY_REQUEST> ]
                   [ <ADMIN_STATUS> ] [ <POLICY_DATA> ... ]
                   <sender descriptor> [<S2L sub-LSP descriptor list>]
                   [<time interval list>]
```

The time interval list in the message is defined below. It is a sequence of TIME-INTERVAL objects, each of which describes a time interval or a series of time intervals.

```
<time interval list> ::=
                   <time interval descriptor>
                   [ <time interval list> ]

<time interval descriptor> ::= <TIME-INTERVAL>
```

7. Behaviors for Temporal LSP

To set up a temporal LSP, the ingress of the LSP MUST include the TIME-INTERVAL objects representing the time intervals configured for the LSP in the PATH message for the LSP.

In addition, the ingress computes a shortest path satisfying the constraints for the LSP in each of the time intervals. it MUST include the ERO for the path in the PATH message for the LSP.

For every node along the path for the LSP, when receiving a PATH message with TIME-INTERVAL objects, it obtains the time intervals represented by the objects in the message received and MUST forward the objects unchanged to the next hop if there is one.

It adds the time intervals into the state for the LSP and checks whether there is enough bandwidth in each of the time intervals. If there is, it reserved the bandwidth on the link to the next hop (if there is a next hop) in each of the time intervals. If there is not, a PathErr message is returned.

8. Security Considerations

The mechanism described in this document does not raise any new security issues for the RSVP-TE protocols.

9. IANA Considerations

This section specifies requests for IANA allocation.

10. Acknowledgement

The author would like to thank people for their valuable comments on this draft.

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, <http://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, <http://www.rfc-editor.org/info/rfc3209>.

[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, <http://www.rfc-editor.org/info/rfc4875>.

[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>.

11.2. Informative References

[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>.

Authors' Addresses

Huaimo Chen
Huawei Technologies
Boston, MA
US

Email: huaimo.chen@huawei.com

Mehmet Toy
Verizon
USA

Email: mehmet.toy@verizon.com

Vic Liu
China Mobile
No.32 Xuanwumen West Street, Xicheng District
Beijing, 100053
China

Email: liu.cmri@gmail.com

Lei Liu
Fijitsu
USA

Email: lliu@us.fujitsu.com

Internet Engineering Task Force
Internet-Draft
Intended status: Experimental
Expires: September 19, 2018

H. Chen, Ed.
Huawei Technologies
R. Torvi, Ed.
Juniper Networks

March 18, 2018

Extensions to RSVP-TE for LSP Ingress FRR Protection
draft-ietf-teas-rsvp-ingress-protection-17.txt

Abstract

This document describes extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for locally protecting the ingress node of a Point-to-Point (P2P) or Point-to-Multipoint (P2MP) Traffic Engineered (TE) Label Switched Path (LSP). It extends the fast-reroute (FRR) protection for transit nodes of an LSP to the ingress node of the LSP. The procedures described in this document are experimental.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

For a MPLS TE LSP, protecting the failures of its transit nodes using fast-reroute (FRR) is covered in RFC 4090 for P2P LSP and RFC 4875 for P2MP LSP. However, protecting the failure of its ingress node using FRR is not covered in either RFC 4090 or RFC 4875. The MPLS Transport Profile (MPLS-TP) Linear Protection described in RFC 6378 can provide a protection against the failure of any transit node of a LSP between the ingress node and the egress node of the LSP, but cannot protect against the failure of the ingress node.

To protect against the failure of the (primary) ingress node of a primary end to end P2MP (or P2P) TE LSP, a typical existing solution is to set up a secondary backup end to end P2MP (or P2P) TE LSP. The backup LSP is from a backup ingress node to backup egress nodes (or node). The backup ingress node is different from the primary ingress node. The backup egress nodes (or node) are (or is) different from the primary egress nodes (or node) of the primary LSP. For a P2MP TE LSP, on each of the primary (and backup) egress nodes, a P2P LSP is created from the egress node to its primary (backup) ingress node and configured with BFD. This is used to detect the failure of the primary (backup) ingress node for the receiver to switch to the backup (or primary) egress node to receive the traffic after the primary (or backup) ingress node fails when both the primary LSP and the secondary LSP carry the traffic. In addition, FRR may be used to provide protections against the failures of the transit nodes and the links of the primary and secondary end to end TE LSPs.

There are a number of issues in this solution:

- It consumes lots of network resources. Double states need to be maintained in the network since two end to end TE LSPs are created. Double link bandwidth is reserved and used when both the primary and the secondary end to end TE LSPs carry the traffic at the same time.

- More operations are needed, which include the configuration of two end to end TE LSPs and BFDs from each of the egress nodes to its corresponding ingress node.

- The detection of the failure of the ingress node may not be reliable. Any failure on the path of the BFD from an egress node to an ingress node may cause the BFD to indicate the failure of the ingress node.

- o The speed of protection against the failure of the ingress node may be slow.

This specification defines a simple extension to RSVP-TE for local protection (FRR) of the ingress node of a P2MP or P2P LSP to resolve these issues. Ingress local protection and ingress FRR protection will be used exchangeably.

Note that this document is experimental. Two different approaches are proposed to transfer the information for ingress protection. They both use the same new INGRESS_PROTECTION object, which is sent in both PATH and RESV messages between a primary ingress and a backup ingress. One approach is Relay-Message Method (refer to section 5.1.1 and 5.2.1), the other is Proxy-Ingress Method (refer to section 5.1.2 and 5.2.2). Each of them has its advantages and disadvantages. It is hard to decide which one is used as a standard approach now. It is expected that the experiment on the ingress local protection with these two approaches provides quantities to help choose one. The quantities include the numbers on control traffic, states, codes and operations. After one approach is selected, the document will be revised to reflect that selection and any other items learned from the experiment. The revised document is expected to be submitted for publication on the standards track.

1.1. Ingress Local Protection Example

Figure 1 shows an example of using a backup P2MP LSP to locally protect the ingress of a primary P2MP LSP, which is from ingress Ia to three egresses: L1, L2 and L3. The backup LSP is from backup ingress Ib to the next hops R2 and R4 of ingress Ia.

```
                      *******  *******              S Source
                   [R2]-----[R3]-----[L1]          Ix Ingress
                  */ &                             Rx Transit
                 */  &                             Lx Egress
                */   &                            *** Primary LSP
               */    &                           &&& Backup LSP across
              */     &                                logical hop
             */      &
            */ ********    ********  *******
      [S]---[Ia]--------[R4]------[R5]-----[L2]
        \      |     &    &           *\
         \     |    &    &             *\
          \    |   &    &               *\
           \   |  &    &                 *\
            \  | &    &                   *\
             \ |&    &                     *\
              [Ib]&&&                       [L3]
```

Figure 1: Ingress Local Protection

In normal operations, source S sends the traffic to primary ingress Ia. Ia imports the traffic into the primary LSP.

When source S detects the failure of Ia, it switches the traffic to backup ingress Ib, which imports the traffic from S into the backup LSP to Ia’s next hops R2 and R4, where the traffic is merged into the primary LSP, and then sent to egresses L1, L2 and L3.

Note that the backup ingress is one logical hop away from the ingress. A logical hop is a direct link or a tunnel such as a GRE tunnel, over which RSVP-TE messages may be exchanged.

1.2. Ingress Local Protection Overview

There are four parts in ingress local protection:

- o Setting up the necessary backup LSP forwarding state based on the information received for ingress local protection;

- o Detecting the primary ingress failure and providing the fast repair (as discussed in Sections 2 and 3);

- o Maintaining the RSVP-TE control plane state until a global repair is done; and

- o Performing the global repair(see Section 5.4.2).

The primary ingress of a primary LSP sends the backup ingress the

information for ingress protection in a PATH message with a new INGRESS_PROTECTION object. The backup ingress sets up the backup LSP(s) and forwarding state after receiving the necessary information for ingress protection. And then it sends the primary ingress the status of ingress protection in a RESV message with a new INGRESS_PROTECTION object.

When the primary ingress fails, the backup ingress sends or refreshes the next hops of the primary ingress the PATH messages without any INGRESS_PROTECTION object after verifying the failure. Thus the RSVP-TE control plane state of the primary LSP is maintained.

2. Ingress Failure Detection

Exactly how to detect the failure of the ingress is out of scope. However, it is necessary to discuss different modes for detecting the failure because they determine what is the required behavior for the source and backup ingress.

2.1. Source Detects Failure

Source Detects Failure or Source-Detect for short means that the source is responsible for fast detecting the failure of the primary ingress of an LSP. Fast detecting the failure means detecting the failure in a few or tens of milliseconds. The backup ingress is ready to import the traffic from the source into the backup LSP(s) after the backup LSP(s) is up.

In normal operations, the source sends the traffic to the primary ingress. When the source detects the failure of the primary ingress, it switches the traffic to the backup ingress, which delivers the traffic to the next hops of the primary ingress through the backup LSP(s), where the traffic is merged into the primary LSP.

For an LSP, after the primary ingress fails, the backup ingress MUST use a method to verify the failure of the primary ingress before the PATH message for the LSP expires at the next hop of the primary ingress. After verifying the failure, the backup ingress sends/refreshes the PATH message to the next hop through the backup LSP as needed. The method may verify the failure of the primary ingress slowly such as in seconds.

After the primary ingress fails, it will not be reachable after routing convergence. Thus checking whether the primary ingress (address) is reachable is a possible method.

When the previously failed primary ingress of a primary LSP becomes

available again and the primary LSP has recovered from its primary ingress, the source may switch the traffic to the primary ingress from the backup ingress. A operator may control the traffic switch through using a command on the source node after seeing that the primary LSP has recovered.

2.2. Backup and Source Detect Failure

Backup and Source Detect Failure or Backup-Source-Detect for short means that both the backup ingress and the source are concurrently responsible for fast detecting the failure of the primary ingress.

Note that one of the differences between Source-Detect and Backup-Source-Detect is: in the former, the backup ingress verifies the failure of the primary ingress slowly such as in seconds; in the latter, the backup ingress detects the failure fast such as in a few or tens of milliseconds.

In normal operations, the source sends the traffic to the primary ingress. It switches the traffic to the backup ingress when it detects the failure of the primary ingress.

The backup ingress does not import any traffic from the source into the backup LSP in normal operations. When it detects the failure of the primary ingress, it imports the traffic from the source into the backup LSP to the next hops of the primary ingress, where the traffic is merged into the primary LSP.

The source-detect is preferred. It is simpler than the backup-source-detect, which needs both the source and the backup ingress detect the ingress failure quickly.

3. Backup Forwarding State

Before the primary ingress fails, the backup ingress is responsible for creating the necessary backup LSPs. These LSPs might be multiple bypass P2P LSPs that avoid the ingress. Alternately, the backup ingress could choose to use a single backup P2MP LSP as a bypass or detour to protect the primary ingress of a primary P2MP LSP.

The backup ingress may be off-path or on-path of an LSP. If a backup ingress is not any node of the LSP, it is off-path. If a backup ingress is a next-hop of the primary ingress of the LSP, it is on-path. When a backup ingress for protecting the primary ingress is configured, the backup ingress MUST not be on the LSP except for it is the next hop of the primary ingress. If it is on-path, the primary forwarding state associated with the primary LSP SHOULD be

clearly separated from the backup LSP(s) state.

3.1. Forwarding State for Backup LSP

A forwarding entry for a backup LSP is created on the backup ingress after the LSP is set up. Depending on the failure-detection mode (e.g., source-detect), it may be used to forward received traffic or simply be inactive (e.g., backup-source-detect) until required. In either case, when the primary ingress fails, this entry is used to import the traffic into the backup LSP to the next hops of the primary ingress, where the traffic is merged into the primary LSP.

The forwarding entry for a backup LSP is a local implementation issue. In one device, it may have an inactive flag. This inactive forwarding entry is not used to forward any traffic normally. When the primary ingress fails, it is changed to active, and thus the traffic from the source is imported into the backup LSP.

4. Protocol Extensions

A new object INGRESS_PROTECTION is defined for signaling ingress local protection. The primary ingress of a primary LSP sends the backup ingress this object in a PATH message. In this case, the object contains the information needed to set up ingress protection. The information includes:

- o Backup ingress IP address indicating the backup ingress,

- o Traffic Descriptor describing the traffic that the primary LSP transports, this traffic is imported into the backup LSP(s) on the backup ingress when the primary ingress fails,

- o Label and Routes indicating the first hops of the primary LSP, each of which is paired with its label, and

- o Desire options on ingress protection such as P2MP option indicating a desire to use a backup P2MP LSP to protect the primary ingress of a primary P2MP LSP.

The backup ingress sends the primary ingress this object in a RESV message. In this case, the object contains the information about the status on the ingress protection.

4.1. INGRESS_PROTECTION Object

4.1.1. Class Number and Class Type

The Class Number for the INGRESS_PROTECTION object MUST be of the form 0bbbbbbb to enable implementations that do not recognize the object to reject the entire message and return an "Unknown Object Class" error [RFC2205]. It is suggested that a Class Number value from the Private Use range (124-127) [RFC3936] specified for the 0bbbbbbb octet be chosen for this experiment. It is also suggested that a Class Type value of 1 be used for this object in this experiment.

The INGRESS_PROTECTION object with the FAST_REROUTE object in a PATH message is used to control the backup for protecting the primary ingress of a primary LSP. The primary ingress MUST insert this object into the PATH message to be sent to the backup ingress for protecting the primary ingress.

4.1.2. Object Format

The INGRESS_PROTECTION object has the following 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Length (bytes)        |    Class-Num  |    C-Type     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Reserved (zero)  |   NUB   |      Flags    |    Options    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                          (Subobjects)                         ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   NUB      Number of Unprotected Branches
   Flags
    0x01    Ingress local protection available
    0x02    Ingress local protection in use
    0x04    Bandwidth protection

   Options
    0x01    Revert to Ingress
    0x02    P2MP Backup
```

For protecting the ingress of a P2MP LSP, if the backup ingress doesn’t have a backup LSP to each of the next hops of the primary ingress, it SHOULD clear "Ingress local protection available" and set NUB to the number of the next hops to which there is no backup LSP.

The flags are used to communicate status information from the backup

ingress to the primary ingress.

- o Ingress local protection available: The backup ingress MUST set this flag after backup LSPs are up and ready for locally protecting the primary ingress. The backup ingress sends this to the primary ingress to indicate that the primary ingress is locally protected.

- o Ingress local protection in use: The backup ingress MUST set this flag when it detects a failure in the primary ingress and actively redirects the traffic into the backup LSPs. The backup ingress records this flag and does not send any RESV message with this flag to the primary ingress since the primary ingress is down.

- o Bandwidth protection: The backup ingress MUST set this flag if the backup LSPs guarantee to provide desired bandwidth for the protected LSP against the primary ingress failure.

The options are used by the primary ingress to specify the desired behavior to the backup ingress.

- o Revert to Ingress: The primary ingress sets this option indicating that the traffic for the primary LSP successfully re-signaled will be switched back to the primary ingress from the backup ingress when the primary ingress is restored.

- o P2MP Backup: This option is set to ask for the backup ingress to use backup P2MP LSP to protect the primary ingress.

The INGRESS_PROTECTION object may contain some subobjects of following 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      |            Length             |Reserved (zero)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Contents/Body of subobject                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where Type is the type of a subobject, Length is the total size of the subobject in bytes, including Type, Length and Contents fields.

4.1.3. Subobject: Backup Ingress IPv4 Address

When the primary ingress of a protected LSP sends a PATH message with an INGRESS_PROTECTION object to the backup ingress, the object MUST

have a Backup Ingress IPv4 Address subobject containing an IPv4 address belonging to the backup ingress if IPv4 is used. The Type of the subobject is 1, and the body of the subobject is given below:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Backup ingress IPv4 address (4 bytes)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Backup ingress IPv4 address: An IPv4 host address of backup ingress

4.1.4. Subobject: Backup Ingress IPv6 Address

When the primary ingress of a protected LSP sends a PATH message with an INGRESS_PROTECTION object to the backup ingress, the object MUST have a Backup Ingress IPv6 Address subobject containing an IPv6 address belonging to the backup ingress if IPv6 is used. The Type of the subobject is 2, the body of the subobject is given below:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Backup ingress IPv6 address (16 bytes)            |
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Backup ingress IPv6 address: An IPv6 host address of backup ingress

4.1.5. Subobject: Ingress IPv4 Address

The INGRESS_PROTECTION object may have an Ingress IPv4 Address subobject containing an IPv4 address belonging to the primary ingress if IPv4 is used. The Type of the subobject is 3. The subobject has the following body:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Ingress IPv4 address (4 bytes)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Ingress IPv4 address: An IPv4 host address of ingress

4.1.6. Subobject: Ingress IPv6 Address

The INGRESS_PROTECTION object may have an Ingress IPv6 Address subobject containing an IPv6 address belonging to the primary ingress if IPv6 is used. The Type of the subobject is 4. The subobject has the following body:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Ingress IPv6 address (16 bytes)                 |
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Ingress IPv6 address: An IPv6 host address of ingress

4.1.7. Subobject: Traffic Descriptor

The INGRESS_PROTECTION object may have a Traffic Descriptor subobject describing the traffic to be mapped to the backup LSP on the backup ingress for locally protecting the primary ingress. The subobject types for Interface, IPv4 Prefix, IPv6 Prefix and Application Identifier are 5, 6, 7 and 8 respectively. The subobject has the following body:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Traffic Element 1                       |
~                                                               ~
|                       Traffic Element n                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Traffic Descriptor subobject may contain multiple Traffic Elements of same type as follows:

- o Interface Traffic: Each of the Traffic Elements is a 32 bit index of an interface, from which the traffic is imported into the backup LSP.

- o IPv4 Prefix Traffic: Each of the Traffic Elements is an IPv4 prefix, containing an 8-bit prefix length followed by an IPv4 address prefix, whose length, in bits, is specified by the prefix length, padded to a byte boundary.

- o IPv6 Prefix Traffic: Each of the Traffic Elements is an IPv6 prefix, containing an 8-bit prefix length followed by an IPv6 address prefix, whose length, in bits, is specified by the prefix length, padded to a byte boundary.

- o Application Traffic: Each of the Traffic Elements is a 32 bit identifier of an application, from which the traffic is imported into the backup LSP.

4.1.8. Subobject: Label-Routes

The INGRESS_PROTECTION object in a PATH message from the primary ingress to the backup ingress may have a Label-Routes subobject containing the labels and routes that the next hops of the ingress use. The Type of the subobject is 9. The subobject has the following body:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                           Subobjects                          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Subobjects in the Label-Routes are copied from those in the RECORD_ROUTE objects in the RESV messages that the primary ingress receives from its next hops for the primary LSP. They MUST contain the first hops of the LSP, each of which is paired with its label.

5. Behavior of Ingress Protection

5.1. Overview

There are two different proposed signaling approaches to transfer the information for ingress protection. They both use the same new INGRESS_PROTECTION object. The object is sent in both PATH and RESV messages.

5.1.1. Relay-Message Method

The primary ingress relays the information for ingress protection of an LSP to the backup ingress via PATH messages. Once the LSP is created, the ingress of the LSP sends the backup ingress a PATH message with an INGRESS_PROTECTION object with Label-Routes subobject, which is populated with the next-hops and labels. This provides sufficient information for the backup ingress to create the

appropriate forwarding state and backup LSP(s).

The ingress also sends the backup ingress all the other PATH messages for the LSP with an empty INGRESS_PROTECTION object. An INGRESS_PROTECTION object without any Traffic-Descriptor subobject is called an empty INGRESS_PROTECTION object. Thus, the backup ingress has access to all the PATH messages needed for modification to refresh control-plane state after a failure.

The empty INGRESS_PROTECTION object is for efficient processing of ingress protection for a P2MP LSP. For a P2MP LSP, its primary ingress may have more than one PATH messages, each of which is sent to a next hop along a branch of the P2MP LSP. The PATH message along a branch will be selected and sent to the backup ingress with an INGRESS_PROTECTION object containing the Traffic-Descriptor subobject; all the PATH messages along the other branches will be sent to the backup ingress containing an INGRESS_PROTECTION object without any Traffic-Descriptor subobject (empty INGRESS_PROTECTION object). For a P2MP LSP, the backup ingress only needs one Traffic-Descriptor.

5.1.2. Proxy-Ingress Method

Conceptually, a proxy ingress is created that starts the RSVP signaling. The explicit path of the LSP goes from the proxy ingress to the backup ingress and then to the real ingress. The behavior and signaling for the proxy ingress is done by the real ingress; the use of a proxy ingress address avoids problems with loop detection. Note that the proxy ingress MUST reside within the same router as the real ingress.

```
                          [ traffic source ]       *** Primary LSP
                           $             $         --- Backup LSP
                           $             $          $$  Link
                           $             $
                    [ proxy ingress ]  [ backup ]
                    [ & ingress     ]     |
                           *              |
                           *****[ MP ]----|
```

Figure 2: Example Protected LSP with Proxy Ingress Node

The backup ingress MUST know the merge points or next-hops and their associated labels. This is accomplished by having the RSVP PATH and RESV messages go through the backup ingress, although the forwarding path need not go through the backup ingress. If the backup ingress

fails, the ingress simply removes the INGRESS_PROTECTION object and forwards the PATH messages to the LSP's next-hop(s). If the ingress has its LSP configured for ingress protection, then the ingress can add the backup ingress and itself to the ERO and start forwarding the PATH messages to the backup ingress.

Slightly different behavior can apply for the on-path and off-path cases. In the on-path case, the backup ingress is a next hop node after the ingress for the LSP. In the off-path, the backup ingress is not any next-hop node after the ingress for all associated sub-LSPs.

The key advantage of this approach is that it minimizes the special handling code required. Because the backup ingress is on the signaling path, it can receive various notifications. It easily has access to all the PATH messages needed for modification to be sent to refresh control-plane state after a failure.

5.2. Ingress Behavior

The primary ingress MUST be configured with a couple of pieces of information for ingress protection.

- o Backup Ingress Address: The primary ingress MUST know the IP address of the backup ingress it wants to be used before it can use the INGRESS_PROTECTION object.

- o Proxy-Ingress-Id (only needed for Proxy-Ingress Method): The Proxy-Ingress-Id is only used in the Record Route Object for recording the proxy-ingress. If no proxy-ingress-id is specified, then a local interface address that will not otherwise be included in the Record Route Object can be used. A similar technique is used in [RFC4090 Sec 6.1.1].

- o Application Traffic Identifier: The primary ingress and backup ingress MUST both know what application traffic should be directed into the LSP. If a list of prefixes in the Traffic Descriptor subobject will not suffice, then a commonly understood Application Traffic Identifier can be sent between the primary ingress and backup ingress. The exact meaning of the identifier should be configured similarly at both the primary ingress and backup ingress. The Application Traffic Identifier is understood within the unique context of the primary ingress and backup ingress.

- o A connection between backup ingress and primary ingress: If there is not any direct link between the primary ingress and the backup ingress, a tunnel MUST be configured between them.

With this additional information, the primary ingress can create and signal the necessary RSVP extensions to support ingress protection.

5.2.1. Relay-Message Method

To protect the primary ingress of an LSP, the primary ingress MUST do the following after the LSP is up.

1. Select a PATH message P0 for the LSP.

2. If the backup ingress is off-path (the backup ingress is not the next hop of the primary ingress for P0), then send it a PATH message P0’ with the content from P0 and an INGRESS_PROTECTION object; else (the backup ingress is a next hop, i.e., on-path case) add an INGRESS_PROTECTION object into the existing PATH message to the backup ingress (i.e., the next hop). The object contains the Traffic-Descriptor subobject, the Backup Ingress Address subobject and the Label-Routes subobject. The options is set to indicate whether a Backup P2MP LSP is desired. The Label-Routes subobject contains the next-hops of the primary ingress and their labels. Note that for on-path case, there is an existing PATH message to the backup ingress (i.e., the next hop), and we just add an INGRESS_PROTECTION object into the existing PATH message to be sent to the backup ingress. We do not send a separate PATH message to the backup ingress for this existing PATH message.

3. For each Pi of the other PATH messages for the LSP, send the backup ingress a PATH message Pi’ with the content copied from Pi and an empty INGRESS_PROTECTION object.

For every PATH message Pj’ (i.e., P0’/Pi’) to be sent to the backup ingress, it has the same SESSION as Pj (i.e., P0/Pi). If the backup ingress is off-path, the primary ingress updates Pj’ according to the backup ingress as its next hop before sending it. It adds the backup ingress to the beginning of the ERO, and sets RSVP_HOP based on the interface to the backup ingress. The primary ingress MUST NOT set up any forwarding state to the backup ingress if the backup ingress is off-path.

5.2.2. Proxy-Ingress Method

The primary ingress is responsible for starting the RSVP signaling for the proxy-ingress node. To do this, the following MUST be done for the RSVP PATH message.

1. Compute the EROs for the LSP as normal for the ingress.

2. If the selected backup ingress node is not the first node on the path (for all sub-LSPs), then insert at the beginning of the ERO first the backup ingress node and then the ingress node.

3. In the PATH RRO, instead of recording the ingress node’s address, replace it with the Proxy-Ingress-Id.

4. Leave the HOP object populated as usual with information for the ingress-node.

5. Add the INGRESS_PROTECTION object to the PATH message. Include the Backup Ingress Address (IPv4 or IPv6) subobject and the Traffic-Descriptor subobject. Set or clear the options indicating that a Backup P2MP LSP is desired.

6. Optionally, add the FAST-REROUTE object [RFC4090] to the Path message. Indicate whether one-to-one backup is desired. Indicate whether facility backup is desired.

7. The RSVP PATH message is sent to the backup node as normal.

If the ingress detects that it can’t communicate with the backup ingress, then the ingress SHOULD instead send the PATH message to the next-hop indicated in the ERO computed in step 1. Once the ingress detects that it can communicate with the backup ingress, the ingress SHOULD follow the steps 1-7 to obtain ingress failure protection.

When the ingress node receives an RSVP PATH message with an INGRESS_PROTECTION object and the object specifies that node as the ingress node and the PHOP as the backup ingress node, the ingress node SHOULD remove the INGRESS_PROTECTION object from the PATH message before sending it out. Additionally, the ingress node MUST store that it will install ingress forwarding state for the LSP rather than midpoint forwarding.

When an RSVP RESV message is received by the ingress, it uses the NHOP to determine whether the message is received from the backup ingress or from a different node. The stored associated PATH message contains an INGRESS_PROTECTION object that identifies the backup ingress node. If the RESV message is not from the backup node, then ingress forwarding state SHOULD be set up, and the INGRESS_PROTECTION object MUST be added to the RESV before it is sent to the NHOP, which SHOULD be the backup node. If the RESV message is from the backup node, then the LSP SHOULD be considered available for use.

If the backup ingress node is on the forwarding path, then a RESV is

received with an INGRESS_PROTECTION object and an NHOP that matches the backup ingress. In this case, the ingress node’s address will not appear after the backup ingress in the RRO. The ingress node SHOULD set up ingress forwarding state, just as is done if the LSP weren’t ingress-node protected.

5.3. Backup Ingress Behavior

An LER determines that the ingress local protection is requested for an LSP if the INGRESS_PROTECTION object is included in the PATH message it receives for the LSP. The LER can further determine that it is the backup ingress if one of its addresses is in the Backup Ingress Address subobject of the INGRESS_PROTECTION object. The LER as the backup ingress will assume full responsibility of the ingress after the primary ingress fails. In addition, the LER determines that it is off-path if it is not any node of the LSP. The LER determines whether it uses Relay-Message Method or Proxy-Ingress Method according to configurations.

5.3.1. Backup Ingress Behavior in Off-path Case

The backup ingress considers itself as a PLR and the primary ingress as its next hop and provides a local protection for the primary ingress. It behaves very similarly to a PLR providing fast-reroute where the primary ingress is considered as the failure-point to protect. Where not otherwise specified, the behavior given in [RFC4090] for a PLR applies.

The backup ingress MUST follow the control-options specified in the INGRESS_PROTECTION object and the flags and specifications in the FAST-REROUTE object. This applies to providing a P2MP backup if the "P2MP backup" is set, a one-to-one backup if "one-to-one desired" is set, facility backup if the "facility backup desired" is set, and backup paths that support the desired bandwidth, and administrative groups that are requested.

If multiple non empty INGRESS_PROTECTION objects have been received via multiple PATH messages for the same LSP, then the most recent one MUST be the one used.

The backup ingress creates the appropriate forwarding state for the backup LSP tunnel(s) to the merge point(s).

When the backup ingress sends a RESV message to the primary ingress, it MUST add an INGRESS_PROTECTION object into the message. It MUST set or clear the flags in the object to report "Ingress local protection available", "Ingress local protection in use", and "bandwidth protection".

If the backup ingress doesn’t have a backup LSP tunnel to each of the merge points, it SHOULD clear "Ingress local protection available" and set NUB to the number of the merge points to which there is no backup LSP.

When the primary ingress fails, the backup ingress redirects the traffic from a source into the backup P2P LSPs or the backup P2MP LSP transmitting the traffic to the next hops of the primary ingress, where the traffic is merged into the protected LSP.

In this case, the backup ingress MUST keep the PATH message with the INGRESS_PROTECTION object received from the primary ingress and the RESV message with the INGRESS_PROTECTION object to be sent to the primary ingress. The backup ingress MUST set the "local protection in use" flag in the RESV message, indicating that the backup ingress is actively redirecting the traffic into the backup P2P LSPs or the backup P2MP LSP for locally protecting the primary ingress failure.

Note that the RESV message with this piece of information will not be sent to the primary ingress because the primary ingress has failed.

If the backup ingress has not received any PATH message from the primary ingress for an extended period of time (e.g., a cleanup timeout interval) and a confirmed primary ingress failure did not occur, then the standard RSVP soft-state removal SHOULD occur. The backup ingress SHALL remove the state for the PATH message from the primary ingress, and tear down the one-to-one backup LSPs for protecting the primary ingress if one-to-one backup is used or unbind the facility backup LSPs if facility backup is used.

When the backup ingress receives a PATH message from the primary ingress for locally protecting the primary ingress of a protected LSP, it MUST check to see if any critical information has been changed. If the next hops of the primary ingress are changed, the backup ingress SHALL update its backup LSP(s) accordingly.

5.3.1.1. Relay-Message Method

When the backup ingress receives a PATH message with an non empty INGRESS_PROTECTION object, it examines the object to learn what traffic associated with the LSP. It determines the next-hops to be merged to by examining the Label-Routes subobject in the object.

The backup ingress MUST store the PATH message received from the primary ingress, but NOT forward it.

The backup ingress responds with a RESV message to the PATH message received from the primary ingress. If the backup ingress is off-

path, the LABEL object in the RESV message contains IMPLICIT-NULL. If the INGRESS_PROTECTION object is not "empty", the backup ingress SHALL send the RESV message with the state indicating protection is available after the backup LSP(s) are successfully established.

5.3.1.2. Proxy-Ingress Method

The backup ingress determines the next-hops to be merged to by collecting the set of the pair of (IPv4/IPv6 subobject, Label subobject) from the Record Route Object of each RESV that are closest to the top and not the Ingress router; this should be the second to the top pair. If a Label-Routes subobject is included in the INGRESS_PROTECTION object, the included IPv4/IPv6 subobjects are used to filter the set down to the specific next-hops where protection is desired. A RESV message MUST have been received before the Backup Ingress can create or select the appropriate backup LSP.

When the backup ingress receives a PATH message with the INGRESS_PROTECTION object, the backup ingress examines the object to learn what traffic associated with the LSP. The backup ingress forwards the PATH message to the ingress node with the normal RSVP changes.

When the backup ingress receives a RESV message with the INGRESS_PROTECTION object, the backup ingress records an IMPLICIT-NULL label in the RRO. Then the backup ingress forwards the RESV message to the ingress node, which is acting for the proxy ingress.

5.3.2. Backup Ingress Behavior in On-path Case

An LER as the backup ingress determines that it is on-path if one of its addresses is a next hop of the primary ingress (and for Proxy-Ingress Method the primary ingress is not its next hop via checking the PATH message with the INGRESS_PROTECTION object received from the primary ingress). The LER on-path MUST send the corresponding PATH messages without any INGRESS_PROTECTION object to its next hops. It creates a number of backup P2P LSPs or a backup P2MP LSP from itself to the other next hops (i.e., the next hops other than the backup ingress) of the primary ingress. The other next hops are from the Label-Routes subobject.

It also creates a forwarding entry, which sends/multicasts the traffic from the source to the next hops of the backup ingress along the protected LSP when the primary ingress fails. The traffic is described by the Traffic-Descriptor.

After the forwarding entry is created, all the backup P2P LSPs or the backup P2MP LSP is up and associated with the protected LSP, the

backup ingress MUST send the primary ingress the RESV message with the INGRESS_PROTECTION object containing the state of the local protection such as "local protection available" flag set to one, which indicates that the primary ingress is locally protected.

When the primary ingress fails, the backup ingress sends/multicasts the traffic from the source to its next hops along the protected LSP and imports the traffic into each of the backup P2P LSPs or the backup P2MP LSP transmitting the traffic to the other next hops of the primary ingress, where the traffic is merged into protected LSP.

During the local repair, the backup ingress MUST continue to send the PATH messages to its next hops as before, keep the PATH message with the INGRESS_PROTECTION object received from the primary ingress and the RESV message with the INGRESS_PROTECTION object to be sent to the primary ingress. It MUST set the "local protection in use" flag in the RESV message.

5.3.3. Failure Detection and Refresh PATH Messages

As described in [RFC4090], it is necessary to refresh the PATH messages via the backup LSP(s). The Backup Ingress MUST wait to refresh the PATH messages until it can accurately detect that the ingress node has failed. An example of such an accurate detection would be that the IGP has no bi-directional links to the ingress node or a BFD session to the primary ingress’ loopback address has failed and stayed failed after the network has reconverged.

As described in [RFC4090 Section 6.4.3], the backup ingress, acting as PLR, MUST modify and send any saved PATH messages associated with the primary LSP to the corresponding next hops through backup LSP(s). Any PATH message sent will not contain any INGRESS_PROTECTION object. The RSVP_HOP object in the message contains an IP source address belonging to the backup ingress. The sender template object has the backup ingress address as its tunnel sender address.

5.4. Revertive Behavior

Upon a failure event in the (primary) ingress of a protected LSP, the protected LSP is locally repaired by the backup ingress. There are a couple of basic strategies for restoring the LSP to a full working path.

- Revert to Primary Ingress: When the primary ingress is restored, it re-signals each of the LSPs that start from the primary ingress. The traffic for every LSP successfully re-signaled is switched back to the primary ingress from the backup ingress.

- Global Repair by Backup Ingress: After determining that the primary ingress of an LSP has failed, the backup ingress computes a new optimal path, signals a new LSP along the new path, and switches the traffic to the new LSP.

5.4.1. Revert to Primary Ingress

If "Revert to Primary Ingress" is desired for a protected LSP, the (primary) ingress of the LSP SHOULD re-signal the LSP that starts from the primary ingress after the primary ingress restores. After the LSP is re-signaled successfully, the traffic SHOULD be switched back to the primary ingress from the backup ingress on the source node and redirected into the LSP starting from the primary ingress.

The primary ingress can specify the "Revert to Ingress" control-option in the INGRESS_PROTECTION object in the PATH messages to the backup ingress. After receiving the "Revert to Ingress" control-option, the backup ingress MUST stop sending/refreshing PATH messages for the protected LSP.

5.4.2. Global Repair by Backup Ingress

When the backup ingress has determined that the primary ingress of the protected LSP has failed (e.g., via the IGP), it can compute a new path and signal a new LSP along the new path so that it no longer relies upon local repair. To do this, the backup ingress MUST use the same tunnel sender address in the Sender Template Object and allocate a LSP ID different from the one of the old LSP as the LSP-ID of the new LSP. This allows the new LSP to share resources with the old LSP. Alternately, the Backup Ingress can create a new LSP with no bandwidth reservation that duplicates the path(s) of the protected LSP, move traffic to the new LSP, delete the protected LSP, and then resignal the new LSP with bandwidth.

6. Security Considerations

In principle this document does not introduce new security issues. The security considerations pertaining to RFC 4090, RFC 4875, RFC 2205 and RFC 3209 remain relevant.

7. Compatibility

This extension reuses and extends semantics and procedures defined in RFC 2205, RFC 3209, RFC 4090 and RFC 4875 to support ingress protection. The new object defined to indicate ingress protection has a class number of the form 0bbbbbbb. Per RFC 2205, a node not

supporting this extension will not recognize the new class number and should respond with an "Unknown Object Class" error. The error message will propagate to the ingress, which can then take action to avoid the incompatible node as a backup ingress or may simply terminate the session.

8. IANA Considerations

This document does not request any IANA actions.

9. Co-authors and Contributors

1. Co-authors

Autumn Liu
Ciena
USA
Email: hliu@ciena.com

Zhenbin Li
Huawei Technologies
Email: zhenbin.li@huawei.com

Yimin Shen
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA
Email: yshen@juniper.net

Tarek Saad
Cisco Systems
Email: tsaad@cisco.com

Fengman Xu
Verizon
2400 N. Glenville Dr
Richardson, TX 75082
USA
Email: fengman.xu@verizon.com

2. Contributors

Ning So
Tata Communications
2613 Fairbourne Cir.
Plano, TX 75082
USA
Email: ningso01@gmail.com

Mehmet Toy
Verizon
USA
Email: mehmet.toy@verizon.com

Lei Liu
USA
Email: liulei.kddi@gmail.com

Renwei Li
Huawei Technologies
2330 Central Expressway
Santa Clara, CA 95050
USA
Email: renwei.li@huawei.com

Quintin Zhao
Huawei Technologies
Boston, MA
USA
Email: quintin.zhao@huawei.com

Boris Zhang
Telus Communications
200 Consilium Pl Floor 15
Toronto, ON M1H 3J3
Canada
Email: Boris.Zhang@telus.com

Markus Jork
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA
Email: mjork@juniper.net

10. Acknowledgement

The authors would like to thank Nobo Akiya, Rahul Aggarwal, Eric Osborne, Ross Callon, Loa Andersson, Daniel King, Michael Yue, Alia Atlas, Olufemi Komolafe, Rob Rennison, Neil Harrison, Kannan Sampath, Gregory Mirsky, and Ronhazli Adam for their valuable comments and suggestions on this draft.

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>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[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>.

[RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090,

DOI 10.17487/RFC4090, May 2005,
<https://www.rfc-editor.org/info/rfc4090>.

[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>.

11.2. 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,
<https://www.rfc-editor.org/info/rfc6378>.

Authors' Addresses

Huaimo Chen (editor)
Huawei Technologies
Boston, MA
USA

Email: huaimo.chen@huawei.com

Raveendra Torvi (editor)
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: rtorvi@juniper.net

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/1id-abstracts.txt

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

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

Copyright Notice

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

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

```
                                  | +---+            __
                                  | |   | TE Node  \/ TTP  o LTP
                                  | +---+
                                  |
                                  | ----- TE Link
                                  | ***** Node Connectivity Matrix,
                                  |       TTP Local Link Connectivity
                                  | @@@@@ TE Tunnel
                                  o-----------------------------------

      Node-1                                                Node-3
   +------------+                                       +------------+
   |    TTP-1   |                                       |    TTP-1   |
   |LTP  __     |            TE-Tunel-1                 |     __     |
   |-6   \/@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\/           |
   o    *  *    oLTP-1           Node-2         LTP-6o    *  *    o
   |   *    *   |            +------------+            |   *    *   |
   |  * TTP-2*  |            |            |            |  * TTP-2*  |
   | *    __   * |LTP-2 LTP-6|            |LTP-1 LTP-5| *    __   * |
   o*    \/    *o-----------o************o-----------o*    \/    *o
   |LTP *  *    | Link-12   |          * | Link-23   |    *  *    |
   |-5 *    *   |      LTP-5|        *   |LTP-2      |   *    *   |
   +--o------o--+           o************o           +--o------o--+
    LTP-4  LTP-3            | *   *    * |            LTP-4  LTP-3
                            |  **     *  |
                            +--o------o--+
                             LTP-4  LTP-3
```

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.

```
                                  |                  |
                                  | +---+           ---
                                  | |   | TE Node  \ / Transitional
                                  | +---+           |  Link
                                  |
                                  | ----- Client Layer Link
                                  | ===== Server Layer Link
                                  | ***** Layer Boundary
                                  o-----------------------------------

      +------------------+
      | +------+         |                           +------+
   -----|Client|------+  |              Client -----|Client|
      | |Layer |---+  |  |              Layer       |Layer |
   -----|Switch|-+ |  |  |              Links  -----|Node  |
      | +------+ | |  |  |                          +------+
      |          | |  |  | Client                    | |  |
      |          | | ---_| Layer                     --- ---
   ***|**********|*| \ /*|****************************\ /*\ /****
      |          ---  |  | Server       Transitional  |   |
      |    Layer \ /  |  | Layer               Links  |   |
      |    Term.  |   |  |                            |   |
      |          |    |  |                            |   |
      |         +------+ |                           +------+
   =============|Server|=====            Server ====|Server|====
      |         |Layer | |               Layer      |Layer |
   =============|Switch|=====            Links  ====|Node  |====
      |         +------+ |                           +------+
      +------------------+

       Physical Node View                       TE-Topology View
```

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.

```
                              | +---+          __
                              | |   | TE Node  \/ TTP  o LTP
                              | +---+
                              |
                              | ----- TE Link
                              | ***** TTP Local Link Connectivity
                              o-----------------------------------

            (IL-1) C-LTP-1 +------------+   C-LTP-2 (IL-1)
                   --------O   (IL-1)   O--------
            (IL-1) C-LTP-3 |   S-TTP-1  |   C-LTP-4 (IL-1)
                   --------O     __     0--------
            (IL-1) C-LTP-5 |    *\/*    |   C-LTP-5 (IL-1)
                   --------O   *    *   O--------
                           |  *(IL-1)*  |
                   S-LTP-3 | * S-TTP-2* |   S-LTP-4
                   --------o*     __    *o--------
                           |    *\/*    |
                           |   *    *   |
                           +--o------o--+
                      S-LTP-1 |      | S-LTP-2
```

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.

```
                                 | +---+
                                 | |   | TE Node
                                 | +---+
                                 | ----- TE Link
                                 o-----------------------------------

     +---+              +---+
     |s31|--------------|S5 |
     +---+\            / +---+
           \          /
            \        /
             \+---+/                +---+
             /|AN1|\----------------|S8 |
            / +---+ \               +---+
     +---+ /         \ +---+
     |S9 |-------------|S11|
     +---+             +---+
           Abstract TE Topology


            +---+                    +---+
            |S1 |--------------------|S2 |
            +---+                    +---+
             /                          \
            /                            \
     +---+ /                  +---+       \ +---+
     |s3 |--------------------|S4 |---------|S5 |
     +---+\                   +---+         +---+
           \                      \             \
            \                      \             \
             \+---+                 +---+         +---+
             /|S6 |\                |S7 |---------|S8 |
            / +---+ \               +---+\       /+---+
     +---+ /         \ +---+              +---+ /
     |S9 |-------------|S10|--------------|S11|/
     +---+             +---+              +---+
           Native 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.

```
                                 | +---+
                                 | |   | TE Node
                                 | +---+
                                 | ----- TE Link
                                 o-----------------------------------

     +---+       +---+        +---+         +---+         +---+
     | R1|-------| R2|--------| R3|---------| R4|---------| R5|
     +---+       +---+        +---+         +---+         +---+
       |                      /   \         /   \         /
       |                     /     \       /     \       /
       |                    /       \     /       \     /
       |                   /         \   /         \   /
       |                  /           \ /           \ /
     +---+             +---+         +---+         +---+
     | R6|-------------| R7|         | R8|---------| R9|
     +---+             +---+         +---+         +---+
```

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.

```
          ---------------
          | Native      |                    |  [ ] TE Node
          | TE-Topology |                    |  +++ TE Link
          ---------------                    o--------------

       [R1] ++++ [R2] ++++ [R3] ++++ [R4] ++++ [R5]
         +                +    +    +    +    +
         +               +      +  +      +  +
         +              +        ++        ++
       [R6] +++++++++ [R7]      [R8] ++++ [R9]
```

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.

```
                                   :
TE info distributed via ISIS-TE    :  TE info distributed via OSPF-TE
                                   :
       +---+       +---+        +---+         +---+         +---+
       | R1|-------| R2|--------| R3|---------| R4|---------| R5|
       +---+       +---+        +---+         +---+         +---+
         |                     / : \         /   \         /
         |                    /  :  \       /     \       /
         |                   /   :   \     /       \     /
         |                  /    :    \   /         \   /
         |                 /     :     \ /           \ /
       +---+             +---+   :    +---+         +---+
       | R6|-------------| R7|   :    | R8|---------| R9|
       +---+             +---+   :    +---+         +---+
                                   :
```

Figure 6a: Example Network Topology

```
 -----------------------   :  -----------------------
 |Native TE Topology    |  :  |Native TE Topology    |
 |Info-Source: ISIS-TE  |  :  |Info-Source: OSPF-TE  |
 -----------------------   :  -----------------------
                           :
  [R1] ++++ [R2] ++++ [R3] : [R3’] ++++ [R4] ++++ [R5]
   +                +      :      +    +    +    +
   +               +       :       +  +      +  +
   +              +        :        ++        ++
  [R6] +++++++++ [R7]      :       [R8] ++++ [R9]
```

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.

```
                                 | +---+            /-\
                                 |   | Router    (   ) WDM
                                 | +---+ Node       \-/  node
                                 |
                                 o----------------------------

   +---+        /-\          /-\           /-\          +---+
   | R1|-------( A )--------( C )---------( E )---------| R3|
   +---+        \-/          \-/           \-/          +---+
                            /   \         /   \
                           /     \       /     \
                          /       \     /       \
                         /         \   /         \
                        /           \ /           \
       +---+          /-\           /-\           /-\          +---+
       | R2|---------( B )---------( D )---------( F )---------| R4|
       +---+          \-/           \-/           \-/          +---+
```

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.

```
                                          | *****  B-F WDM Path
                                          | @@@@@  B-E WDM Path
                                          | $$$$$  A-E WDM Path
                                          o--------------------

   +---+        /-\ $$$$$$$$ /-\ $$$$$$$$$ /-\          +---+
   | R1|-------( A )--------( C )---------( E )---------| R3|
   +---+        \-/         @\-/ @@@@@@@@@ \-/          +---+
                            @/   \         /   \
                           @/     \       /     \
                          @/       \     /       \
                         @/         \   /         \
                        @/           \ /           \
       +---+          /-\ ********* /-\ ********* /-\          +---+
       | R2|---------( B )---------( D )---------( F )---------| R4|
       +---+          \-/           \-/           \-/          +---+
```

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.

```
     /---\      +---+    +---+      +---+    +---+      /---\
     |s3 |------|S13|----|S15|------|S23|----|S25|------|C21|
     \---/      +---+\   +---+      +---+   /+---+      \---/
                      \                    /
                       \                  /
                        \+---+      +---+/   +---+      /---\
                         |S18|------|S24|    |S28|------|C22|
                         +---+      +---+\  /+---+      \---/
                                           \/
                                           /\
     /---\      +---+    +---+      +---+/  \+---+      /---\
     |C12|------|S19|----|S17|------|S29|----|S27|------|C23|
     \---/      +---+    +---+      +---+    +---+      \---/


        Domain 1 TE Topology              Domain 2 TE Topology

             +---+    +---+                 +---+    +---+
        -----|S13|----|S15|----          ----|S23|----|S25|----
             +---+\   +---+                 +---+   /+---+
                   \                               /
                    \                             /
                     \+---+                 +---+/   +---+
                      |S18|----          ----|S24|    |S28|----
                      +---+                 +---+\  /+---+
                                                 \/
                                                 /\
             +---+    +---+                 +---+/  \+---+
        -----|S19|----|S17|----          ----|S29|----|S27|----
             +---+    +---+                 +---+    +---+
```

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

         +---+    +---+                 +---+    +---+
   -----|S13|----|S15|----          ----|S23|----|S25|----
         +---+\   +---+                 +---+   /+---+
               \                               /
                \                             /
                 \+---+                 +---+/   +---+
                  |S18|----          ----|S24|    |S28|----
                  +---+                 +---+\  /+---+
                                              \/
                                              /\
         +---+    +---+                 +---+/  \+---+
   -----|S19|----|S17|----          ----|S29|----|S27|----
         +---+    +---+                 +---+    +---+


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

         +------------+                 +------------+
   -----|            |----          ----|            |----
         |            |                 |            |
         |    AN1     |----          ----|    AN1     |----
         |            |                 |            |
   -----|            |----          ----|            |----
         +------------+                 +------------+
```

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.

Client's Merged TE Topology 2

```
/---\      +------------+      +------------+      /---\
|s3 |------|            |------|            |------|C21|
\---/      |            |      |            |      \---/
           |            |      |            |
           |            |      |            |
           |            |      |            |      /---\
           |    AN11    |------|    AN21    |------|C22|
           |            |      |            |      \---/
           |            |      |            |
           |            |      |            |
/---\      |            |      |            |      /---\
|C12|------|            |------|            |------|C23|
\---/      +------------+      +------------+      \---/
```

Client's Merged TE Topology 3

```
/---\      +------------+      +---+    +---+      /---\
|s3 |------|            |------|S23|----|S25|------|C21|
\---/      |            |      +---+   /+---+      \---/
           |            |             /
           |            |            /
           |            |      +---+/   +---+      /---\
           |    AN11    |------|S24|    |S28|------|C22|
           |            |      +---+\  /+---+      \---/
           |            |            \/
           |            |            /\
/---\      |            |      +---+/  \+---+      /---\
|C12|------|            |------|S29|----|S27|------|C23|
\---/      +------------+      +---+    +---+      \---/
```

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].

```
+-------------------------+
|                         |
| Network Topology Model  |
| (ietf-network-topology)|
+-------------------------+
             |
             |
             |
             V
+-------------------------+
|       TE Topology       |
|         Model           |
|                         |
+-------------------------+
```

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.

```
                  +-----------------------------+
                  |      TE Topology Model      |
                  | (Defined in This Document)  |
                  +-----------------------------+
                                 |
            +-------------+-------------+-------------+
            |             |             |             |
            V             V             V             V
      +------------+                            +------------+
      | Technology |                            | Technology |
      |  Specific  |   ......................   |  Specific  |
      | TE Topology|                            | TE Topology|
      |   Model 1  |                            |   Model n  |
      +------------+                            +------------+
```

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
        ...........
        |  |  +--ro 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 admin-status?            te-types:te-admin-status
      |  |  ....................
      |  +--rw underlay-topology {te-topology-hierarchy}?
      |     +--rw network-ref?   leafref

augment /nw:networks/nw:network/nt:link:
   +--rw te!
      +--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
         "Describining 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
                   destinatioin 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 terminiation 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:

- /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.

- /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.

- /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.

- /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.

- /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.

- /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] Shiomoto, 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., Shiomoto, 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 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 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 generic?   int32
          |  |      |        +--rw range-bitmap?   yang:hex-string
          |  |      +--rw to
          |  |      |  +--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 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-
   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 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?
          |  |  |     |     |             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?
          |  |     |  |       -> /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-
   label
          |  |  |  |                +--ro direction?
          |  |  |  |                        te-label-direction
          |  |  |  +--ro backup-path* [index]
          |  |  |  |  +--ro index           uint32
          |  |  |  |  +--ro network-ref?
          |  |  |  |  |       -> /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-
   label
          |  |  |  |                 +--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
          |   |  |     |  |  |        |     +--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
```

```
          |  |  |     |  |           |     |       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 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
```

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

```
          |  |      |        +--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 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-
   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 is-allowed?         boolean
          |  |     +--ro underlay {te-topology-hierarchy}?
          |  |     |  +--ro enabled?                     boolean
          |  |     |  +--ro primary-path
          |  |     |  |  +--ro network-ref?
          |  |     |  |  |       -> /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-label
           |  |      |  |                 +--ro direction?
           |  |      |  |                         te-label-direction
           |  |      |  +--ro backup-path* [index]
           |  |      |  |  +--ro index           uint32
           |  |      |  |  +--ro network-ref?
           |  |      |  |  |       -> /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-label
          |  |      |  |                 +--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
```

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

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

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

```
          +--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

Network Working Group K. Kumaki
Internet-Draft T. Miyasaka
Intended status: Informational KDDI Corporation
Expires: January 7, 2016 July 6, 2015

ACTN : Use case for Multi Tenant VNO
draft-kumaki-teas-actn-multitenant-vno-00

Abstract

This document provides a use case that addresses the need for facilitating virtual network operation: creation and operation of multi-tenant virtual networks that use the common core network resources. This will accelerate a rapid service deployment of new services, including more dynamic and elastic services, and improve overall network operations and scaling of existing services. This use case addresses the aforementioned needs within a single operator network.

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 7, 2016.

Table of Contents

1. Introduction

This document provides a use case that addresses the need for facilitating virtual network operation: creation and operation of multi-tenant virtual networks that use the common core network resources. This will accelerate a rapid service deployment of new services, including more dynamic and elastic services, and improve overall network operations and scaling of existing services.

Generally, carrier’s core network consists of multi-domain networks. For instance, a network may have multiple confederation ASes that have each own IGP network using IS-IS or OSPF. In other instance, a network may consist of multiple vendor’s optical transport networks that have each own Network Management System (NMS). In such a carrier’s multi-domain core network, it is difficult to create an end-to-end virtual network that meets customer’s requirements (e.g. topology, AS, bandwidth, delay and jitter).

This use case supports Abstraction and Control of Transport Networks (ACTN). The aim of ACTN is to facilitate virtual network operation, creation of a virtualized environment allowing operators to view and control multi-subnet multi-technology networks into a single virtualized network. Related documents are: [I-D.leeking-teas-actn-problem-statement] and [I-D.ceccarelli-teas-actn-framework] which provide detailed information regarding this work.

2. 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].

3. Motivation

One of the main motivations for multi-tenant virtual networks that share the common core transport network resource is to increase the network utilization of the core transport network. As each service network has evolved in a different time with different service needs, many dedicated overlay networks have formed to support different service needs. This results in an inefficient use of network resources and the complexity in operating such diverse service networks. Due to the lack of the coordination across different service networks and the common service platform, the introduction of new services is not as speedy as the operators’ desire. Part of the reasons for this difficulty is due to the lack of the virtual network infrastructure. Figure 1 shows an illustration of the current multiple service network architecture.

```
   +-----------+                                +-----------+
   | Service A |                                | Service A |
   |  Network  |\                             /|  Network  |
   |           | \                           / |           |
   +-----------+  \+-------------------------+/  +-----------+
                   |                           |
   +-----------+   |     Core Transport        |   +-----------+
   | Service B |   |         Network           |   | Service B |
   |  Network  |---|                           |---|  Network  |
   |           |   |                           |   |           |
   +-----------+   |                           |   +-----------+
                   +-------------------------+
   +-----------+  /                            \  +-----------+
   | Service C | /                              \ | Service C |
   |  Network  |/                                \|  Network  |
   |           |                                  |           |
   +-----------+                                  +-----------+
```

Figure 1: Multiple Services Network Architecture

The characteristics of the multiple services network are as follows:

- o Each service has its own dedicated access points (e.g., PE routers) in the core network.

- o Each service or a group of services may be operated in a different service operations department within an operator. For instance, the VPN service and the mobile service may be operated by two different departments while whole sale Internet service by another department.

- o There may be dedicated core transport network resources for some services to ensure a strict service guarantee.

- o There may be little or no coordination for operating multiple services in terms of network resource allocation or sharing of the resources.

4. Multi-tenant Virtual Network Consolidation

This section discusses key aspects to support multi-tenant virtual network consolidation.

4.1. Service Consolidation

Multi-tenant virtual network operation should support different services as the tenants that share the common core transport network

resources. Therefore, it is important to understand the type of various services and its service requirement.

4.2. VPN Service Consolidation

Network providers have many different service networks such as VPNs of various types and different QoS requirements. Within VPNs, there are several QoS levels. Some VPN is best-effort VPN while other VPNs require a strict QoS such as bandwidth guarantee and latency. Therefore, multi-level VPNs should be supported in multi-tenant virtual network consolidation.

4.3. Network Wholesale Service

Network providers want to provide a network resource (i.e. a network slice) to ISPs. Generally, ISPs have each own ASes and they do not want to change their own network policy. We can deploy these network wholesale services by using "carrier’s carrier" in [RFC4364]. However, ISPs need to change their network policy and run LDP at an edge router at ISP’s edge routers although just IP is running on their original network. Additionally, they (i.e. network architects at ISPs) need to transfer MPLS knowledge to their network operators.

In another aspect, the network provider must guarantee the SLA to each ISP. There may be different level of SLA as well as different level of virtual network granularity for each ISP. The ISP should be given its virtual network(s) as well as an independent domain control of allocated virtual network(s). It is also to be noted that there may be different grade of services required depending on the nature of the whole sale. For instance, CATV operator may require a different grade of service than best-effort internet services. Therefore, multi-level wholesale services should be supported in multi-tenant virtual network consolidation. Also, network providers should not provide unnecessary network information (e.g. TE database and IGP information in core transport network) to ISPs. To provide unnecessary information in core transport network poses security issues. Therefore, network providers should provide only necessary network information to create ISP’s virtual network.

4.4. On-demand Network Service

Some ISPs may need a network resource (i.e. a network slice) during the specific time and period. This is referred to as on-demand network service. This implies that virtual networks should be created/deleted dynamically and the resources (e.g. bandwidth) of virtual networks should be added/decreased dynamically.

4.5. Redundant Network Service

Some service requires a number of redundant network paths that are physically diverse from one another. This implies that the virtual networks should indicate link and node diversity constraints.

4.6. Mobile/LTE Access Service

Consumer mobile/LTE access can be a tenant that shares the resources of the core transport network. In such case, a strict latency with a guaranteed bandwidth should be supported by multi-tenant virtual network operation.

5. Multi-tenant Virtual Network Operation Coordination

The following Figure 2 depicts a functional control architecture that shows the need to support virtual networks to a number of different service networks that share the common core network resources.

```
                        +-----------------+
                        |  Multi-tenant   |
                        | VN Coordination |
   +-----------+        +-----------------+
   | Service A |-+           |   |
   |  Control  |B|-+         |   |
   +-----------+ |C|---------|   |
     +-----------+ |             |
       +-----------+     +---------------+
                         |Core Transport |
          /------\       |Network Control|      /------\
        //        \\     +---------------+    //        \\
       |  Service A |            |           |  Service A |
        \\        //        ----------        /\        //
           ------   \   ////          \\\\   /   ------
       /------\       \||                  ||/       /------\
     //        \\     |     Core Transport   |     //        \\
    |  Service B |----|                      |----|  Service B |
     \\        //      ||       Network    ||      \\        //
       \------/          \\\\          ////          \------/
          /------\    /     ----------    \     /------\
        //        \\ /                      \  //        \\
       |  Service C |                        \|  Service C |
        \\        //                          \\        //
           ------                                ------
```

Figure 2: Multi-tenant control architecture

There are a few characteristics of the above architecture.

1. The core transport network is the common transport network resource pool for a number of multiple tenants, which is referred to as network tenancy.

2. Each service is a client to the common transport network.

3. Each service should be guaranteed its operational independence from other services. The separation of service control (depicted as separate boxes) in the above figure represents an operational independence.

4. The virtual network for each service is created and assigned by the multi-tenant virtual network coordination function. This is a functional entity that communicates with each service control and the core transport network control/management entities in order to coordinate with the necessary communication.

5. Each service instantiates its service instance based on its virtual network.

6. Each service is in control of its virtual network and operates on the virtual network.

7. As a number of services carried on the common transport network sharing a common network resource, operational independence for each service has to be guaranteed as if each service owns its dedicated virtual resources.

8. The level of abstraction of a virtual network is determined by each service and may differ from one another. In some cases, a virtual network should represent a graph form of topology abstraction of the virtual network.

6. High-level Requirements for Multi-tenant Virtual Network Operations

Based on the discussion in the previous sections, this section provides the overall requirements that must be supported.

6.1. Dynamic binding - On-demand Virtual Network Service Creation

The solution needs to provide the ability to create a new virtual network on demand. The virtual network should be built dynamically.

6.2. Domain Control Plane/Routing Layer Separation

The solution needs to support an independent control plane for a domain service control. This implies that each service domain has its own VN control scheme that is independent of other domain or the core transport network control.

6.3. Separate Operation of Virtual Services

The solution needs to support an independent operation of a virtual network and a service. Each Service Administrators should be able to control and manage its virtual network in terms of policy and resource allocation (e.g., CPU, Memory, other resources.) In addition, the virtualized networks should not affect each other in any way.

6.4. QoS/SLA

The solution needs to provide an independent QoS/SLA per a virtual network depending on a service level. Each QoS on the virtual network should support multiple service levels. Each SLA on the virtual network should fulfill a bandwidth and a latency required by each service.

6.5. VN diversity

Each service should be able to create multiple diverse VNs for the diversity purpose. The diversity for VNs must be physically diverse in the core transport network. This implies that the core transport network control/management plane must be able to factor the SRLG information when creating multiple VNs to ensure VN diversity.

6.6. Security Concerns

The solution needs to keep the confidentiality between the services. A service should not have the connectivity to an another service through the common core transport network.

7. Acknowledgments

The authors wish to thank Young Lee and Dhruv Dhody for the discussions in the document.

8. IANA Considerations

9. Security Considerations

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.

10.2. Informational References

[I-D.ceccarelli-teas-actn-framework]
Ceccarelli, D. and Y. Lee, "Framework for Abstraction and Control of Transport Networks", draft-ceccarelli-teas-actn-framework-00 (work in progress), June 2015.

[I-D.leeking-teas-actn-problem-statement]
Lee, Y., King, D., Boucadair, M., Jing, R., and L. Contreras, "Problem Statement for Abstraction and Control of Transport Networks", draft-leeking-teas-actn-problem-statement-00 (work in progress), June 2015.

Authors' Addresses

Kenji Kumaki
KDDI Corporation
1-20-1 Nishishinjuku
Shinjuku-ku, Tokyo 160-0023
Japan

Email: ke-kumaki@kddi.com

Takuya Miyasaka
KDDI Corporation
1-20-1 Nishishinjuku
Shinjuku-ku, Tokyo 160-0023
Japan

Email: ta-miyasaka@kddi.com

TEAS Working Group
Internet Draft
Intended status: Informational
Intended status: Informational
Expires: April 2017

K. Lam
E. Varma
Nokia
P. Doolan
Coriant
N. Davis
Ciena
B. Zeuner
Deutsche Telekom
M. Betts
ZTE
I. Busi
Huawei
S. Mansfield
Ericsson
R. Vilalta
CTTC
V. Lopez
Telefonica

October 27, 2016

Usage of IM for network topology to support TE Topology YANG Module Development

draft-lam-teas-usage-info-model-net-topology-04.txt

Status of this Memo

Abstract

The benefits of using a common Information Model (IM) as a foundation for deriving purpose and protocol specific interfaces, particularly

for complex networking domains, has been described in draft-betts-netmod-framework-data-schema-uml. This draft describes existing information model relevant to Network Topology and illustrates how it can be used to help ensure the consistency and completeness of the YANG data modeling for TE topologies solutions work in TEAS.

Table of Contents

1. Introduction

This draft describes existing information modeling (IM) relevant to Network Topology [ONF TR-512] [OSSDN SNOWMASS] and illustrates how it can be used to help ensure the consistency and completeness of the YANG data model (DM) for TE topologies solutions development work in TEAS.

2. Background and Motivation

Information Models (IM) and Data Models (DM) are related but different. An IM provides an abstract, conceptual view of the system being modeled in terms of its constituent parts (objects), independent of any specific implementations or protocols used to transport the data; it hides all protocol and implementation details (RFC 3444, TM Forum/NGCOR, ITU-T SG 15). A DM is a concrete specification in a particular language of an interface to, in this case, a controlled/managed system. The intention of the distinction between IMs and DMs has been to separate the modeling of problem space semantics from the modeling of the implementation of those semantics (though the dividing line has not always been clearly articulated).

A DM may be derived from an IM though it is often created without (explicit or obviously implicit) reference to one. When a DM is

derived from an IM, the DM and the components of the system it provides control/management access to are traceable to the definitions provided in the IM. There is no ambiguity between designer, developer, user or operator regarding the name, function, and information elements that are associated with a particular managed object.

As described in [I-D.betts], when DMs are created "in isolation" solely for the purpose of encoding specific interfaces, they may do that job adequately for any particular interface but in complex domains may create opportunities for confusion, duplication of effort, lack of interoperability, and lack of extensibility. In the past, ad-hoc development of DMs has caused significant operational and implementation inefficiencies in our industry.

Since March 2014, upon IESG recommendation that SNMP no longer be used for new work re configuration and that NETCONF/YANG be used instead, there has been an explosion of YANG DM development in IETF. It has consequently been recognized as essential to assure proper coordination of YANG DM development (including reaching out to different SDOs/consortia), as well as to assure that the YANG modules themselves provide a good representation of what is being modeled, to meet expectations of functionality, quality, and interoperability. In order to facilitate this objective, guidance from available pertinent IMs can be valuable.

This draft first describes an existing information model relevant to Network Topology [ONF TR-512], which is part of the Common Information Model (ONF-CIM) of network resources (as described in [I-D.betts]), that can be leveraged to assess the consistency and completeness of related YANG modules under development. It also describes an transport application-specific IM [OSSDN SNOWMASS], derived from CIM pruning and refactoring as explained in [I-D.betts], that is intended to enable further clarity in understanding the modeling. Being part of a Common Information Model, it will not lead to development of incompatible/uncoordinated models that can be difficult to maintain as other purpose-specific interfaces are developed.

3. The Common Information Model

This section provides a high level introduction to the ONF Common Information Model (ONF-CIM), and in particular its Core Model (see [ONF TR-512]), to provide an overall context for the topology relevant subset. The ONF-CIM has been developed through collaboration among several SDOs, including ITU-T, TM Forum, and ONF, and also published as ITU-T Recommendation G.7711 [G.7711].

An information model describes the things in a domain in terms of objects, their properties (represented as attributes), and their relationships.

The ONF-CIM is expressed in a formal language called UML (Unified Modeling Language). UML has a number of basic model elements, called UML artifacts. In order to assure a consistent and harmonized modeling approach, only a selected subset of these UML artifacts were used in the development of the ONF-CIM according to guidelines for creating an information model expressed in UML (see the UML Guidelines document in the ONF TR-514 [ONF TR-514]).

The ONF-CIM has been developed using the Papyrus open source UML Tool, for which a detailed guidelines document is available (see the Papyrus Guidelines document in the ONF TR-515 [ONF TR-515]). This guidelines document also describes how the modelers constructing the ONF-CIM can cooperate in the GitHub environment to allow for separate and still coordinated development of the ONF-CIM fragments.

The OMF-CIM includes all of the artifacts (objects, attributes, associations, etc.) that are necessary to describe the domain for the applications being developed.

It will be necessary to continually expand and refine the ONF-CIM over time as, for example to add, new applications, capabilities or forwarding technologies, or to refine the ONF-CIM as new insights are gained. To allow these extensions to be made in a seamless manner, the ONF-CIM is structured into a number of sub-models. This modeling approach enables application specific and forwarding technology specific extensions to be developed by domain experts with appropriate independence. This approach is further articulated in ONF TR-513 [ONF TR-513] and [I-D.betts].

3.1. Core Model

The Core Model of the ONF-CIM consists of model artifacts that are intended for use by multiple applications and/or forwarding technologies.

For navigability, the Core Model is further sub-structured into sub-models. Currently, these consist of the Core Network Model (CNM), Core Foundation Model, Core Physical Model, and the Core Specification Model. The following sub-sections provide an overview of these sub-models. A detailed description is contained in ONF TR-512 [ONF TR-512].

3.1.1. Core Network Model

The Core Network Model (CNM) consists of artifacts that model the essential network aspects that are neutral to the forwarding technology of the network. The CNM currently encompasses Topology, Termination, and Forwarding aspects (subsets of the CNM) as described below:

- Topology Subset of CNM

  The Topology subset of the CNM supports the modeling of network topology information, which can be used to build the topology database and depict the topology. Object classes representing topological entities include:

  o Forwarding Domain (FD): Offers the potential to enable forwarding of information.

  o Link (L): Models the adjacency between two or more FDs. A Link has LinkPorts.

  o Logical Termination Point (LTP): Models the ports of a link. It encapsulates the termination, adaptation, and OAM functions of one or more transport layers.

  o Network Element (NE): While not actually part of topology, a NE brings meaning to the FD and the LTP contexts (and hence the links). A NE represents physical equipment "bundling" to

provide a view of management scope, management access, and session.

The Topology subset of the CNM supports network topology abstraction and virtualization. FD abstraction is supported via recursive aggregation and virtualization via partitioning of resources according to the resource dedication criterion.

- Forwarding Subset of CNM

  The Forwarding subset of the CNM (not covered in detail in this draft) supports configuration of forwarding entities, including their setup, modification, and tear down. Artifacts representing the forwarding construct include:

  o ForwardingConstruct (FC): Also known as SNC. In conjunction with the FcPort, FC models the enabled forwarding between two FcPorts across a FD.

  o FcPort: Models the access to the FC, and associates the FC to the LTP. When the FC supports protection, the FcPort also indicates its role in the protection scheme, i.e., whether it is a working or protection FcPort.

  o FcRoute: Also known as SncRoute. It models the individual routes of an FC.

  o FcSwitch: Also known as SncSwitch. It models the switched forwarding of traffic (traffic flow) between EPs and is present where there is protection functionality in the FD.

- Termination Subset of CNM

  The Termination subset of the CNM (not covered in detail in this draft) supports modeling of the processing of transport characteristic information, such as termination, adaptation, OAM, etc. Artifacts representing the termination and adaptation and OAM construct include:

  o Logical Termination Point (LTP): See the LTP description in the Topology Subset

o Layer Protocol (LP): This identifies the type of signal and is the anchor for transport layer protocol specific definitions, which are modeled in, e.g., [G.874.1] for OTN, [G.8052] for transport Ethernet, and [G.8152] for MPLS-TP.

- Resilience Subset of CNM

  The Resilience subset provides a view of the model for resilience (including protection and restoration) and encompasses:

  o The basic resilience model structure

  o The key attributes relevant to resilience

  o The application of the resilience model to various cases

3.1.2. Core Foundation Model

To communicate about an entity, it is important to have some way of referring to that entity, i.e., to have some way of referencing it. The Core Foundation model defines the artifacts for referencing entities; i.e.:

- Global Unique ID (GUID):

  An identifier that is globally unique where an identifier is a property of an entity/role with a value that is unique within an identifier space, where the identifier space is itself unique, and immutable. The identifier therefore represents the identity of the entity/role. An identifier carries no semantics with respect to the purpose of the entity.)

- Local ID:

  An identifier that is unique in the context of some scope that is less than the global scope (where an identifier is as defined in GUID above).

- Name:

A property of an entity with a value that is unique in some namespace but may change during the life of the entity. A name carries no semantics with respect to the purpose of the entity.

- Label:

  A property of an entity with a value that is not expected to be unique and is allowed to change. A label carries no semantics with respect to the purpose of the entity and has no effect on the entity behavior or state.

The Core Foundation model also provides the opportunity to extend any entity using the Extension structure.

The model also defines two foundation object classes:

- GlobalClass:

  Super class of object classes for which their instances can exist on their own right, e.g. NE, LTP, FD, Link, and FC. Global classes shall have one and only one globally unique identifier (GUID) and may have zero or more local identifiers, zero or more names, zero or more labels, zero or more extensions.

- LocalClass:

  Super class of object classes for which the existence of their instances depends on instances of global classes; e.g., LP (of LTP), EP (of FC), and LE (of Link). Local classes shall have at least one local identifier, may have zero or more names, zero or more labels, zero or more extensions.

```
------------------------------------------------------------------
I                                                                I
I                                                                I
I                                                                I
I                                                                I
I             Artifacts for Referencing of Entities              I
I                                                                I
```

```
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                     (only in PDF version)                        I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   ---------------------------------------------------------------------
```

Figure 3-1 Artifacts for Referencing of Entities

The Core Foundation model also defines a State_Pac artifact, which is a package of state attributes. The State_Pac is inherited by GlobalClass and LocalClass object classes. The State_Pac consists of the following state-related attributes:

- Operational State:

  Read-only with values: DISABLED, ENABLED

- Administrative State:

  Read-write with values: LOCKED, UNLOCKED

- Lifecycle State:

  Read-write with values: PLANNED, POTENTIAL, INSTALLED, PENDING_REMOVAL

```
   ---------------------------------------------------------------------
   I                                                                  I
   I                                                                  I
```

```
   I                                                                  I
   I                        States of Objects                         I
   I                                                                  I
   I                                                                  I
   I                      (only in PDF version)                       I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   ---------------------------------------------------------------------
```

Figure 3-2 States of Objects

3.1.3. Core Physical Model

The Physical model provides a view of the model for physical entities (including equipment, holders and connectors). This model also specifies the relationship between the connector and the LTP, and the relationship between physical and functional views.

```
   ---------------------------------------------------------------------
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                        Basic Equipment Pattern                   I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                      (only in PDF version)                       I
   I                                                                  I
```

```
   I                                                                  I
   I                                                                  I
   I                                                                  I
   I                                                                  I
   ---------------------------------------------------------------------
```

Figure 3-3 Basic Equipment Pattern

3.1.4. Core Specification Model

There are several related needs that have given rise to the Specification model:

- Provide machine readable form of specific localized behavior:

  o Representing rules related to restrictions of specific cases of use of the model

  o Representing capabilities of specific cases of use

- Enable the introduction of run time schema where the essential structure of the model is known up front (at compile time) but the details are not

- Reduce the clutter in a representation where a set of details take the same values for all instances that are related to a specific case

- Allow leverage of existing standards definitions (e.g., technology/application specific) in a machine readable language

The combination of the above resulted in a separation in the model of definitions of structure and content such that an instance of a class from one model fragment could have an association instance to another model fragment to enable the provision of a fragment of definition of the class and of subordinates.

The aim of all specification definitions is that they be rigorous definitions of specific cases of usage and enable machine

interpretation where traditional interface designs would only allow human interpretation.

The following dedicated spec structures have been considered:

- FC spec: Main focus to provide a representation of the effective internal structure of a ForwardingConstruct (FC)

- LTP and LP spec: Main focus to provide a representation of Layer Protocol (LP) specific parameters for the Logical Termination Point (LTP)

- FD and Link spec: Main focus on capacity and forwarding enablement restrictions

- Equipment spec: Main focus to provide a representation of equipping constraints

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I          Class Diagram of the Spec Model of LTP and LP            I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                      (only in PDF version)                        I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   ---------------------------------------------------------------------
```

Figure 3-4 Class Diagram of the Spec Model of LTP and LP

3.2. Other Models

In addition to the Core Model, the ONF-CIM includes forwarding technology and application specific models. The forwarding technology models of the ONF-CIM (see [ONF TR-512]) encompasses transport technology layers 0, 1, and 2.

4. High Level Description of the Topology Subset of the CNM

This section provides a high-level overview of the Topology Subset of the CNM. Figure 4-1 below is a skeleton class diagram illustrating the key object classes. To avoid cluttering the figure, not all associations have been shown and all of the attributes were omitted.

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I               Overview of the CNM Topology Subset                 I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                     (only in PDF version)                         I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   ---------------------------------------------------------------------
```

Figure 4-1 Overview of the CNM Topology Subset

4.1. Object Classes of the CNM Topology Subset

This section describes the object classes of the Topology Subset of the CNM. Relationships between these classes are described in section 4.2 below

4.1.1. LogicalTerminationPoint (LTP) and LayerProtocol (LP)

The LogicalTerminationPoint (LTP) object class encapsulates the termination, adaptation and OAM functions of one or more transport protocol layers. The structure of the LTP supports all transport protocols including circuit and packet forms. Each transport layer is represented by a LayerProtocol (LP) instance. The LayerProtocol instances of the LTP can be used for controlling the termination and OAM functionality of that layer. It can also be used for controlling the adaptation (i.e. encapsulation and/or multiplexing of client signal). Where the client/server relationship is fixed 1:1 and immutable, the different layers can be encapsulated in a single LTP instance. Where there is a n:1 relationship between client and server, the layers must be split over separate instances of LTP.

The LP object class is defined with generic attributes "layerProtocolName" for indicating the supported transport layer protocol.

Transport layer specific properties (such as layer-specific termination and adaptation properties) are modeled as attributes of conditional packages (called "_Pacs" in the UML notation of the ONF-CIM) associated with the LP object class.

4.1.2. ForwardingDomain (FD)

The ForwardingDomain (FD) object class models the switching and routing capabilities (see "subnetwork" topological component in [G.852.2] and [TMF612]), which is used to effect forwarding of transport characteristic information and offers the potential to enable forwarding. It represents the resource that supports flows across the FD. The FD object can hold zero or more instances of ForwardingConstruct (FC) (representing constrained forwarding, not discussed further in this document, covering connections, VLANs etc) of one or more layer networks; e.g., OCh, ODU, ETH, and MPLS-TP. The

FD object provides the context for operations that create/modify/delete FCs.

The FD object class supports a recursive aggregation relationship such that the internal construction of an FD can be exposed as multiple lower level FDs and associated Links (partitioning) (see section 4.2.1.)

At the lowest level of recursion, a FD (within a network element) could represent a switch matrix (i.e., a fabric).

Note that an NE can encompass multiple switch matrices (FDs), as described in section 4.2.2. An instance of FD is associated with zero or more LTP objects, as described in section 4.2.3.

4.1.3. Link and Link Port

The Link object class models the adjacency between two or more ForwardingDomains (FDs).

In its basic form (i.e., point-to-point Link) it associates a set of LTP clients on one FD with an equivalent set of LTP clients on another FD. Like the FC, the Link has endpoints (LinkPort) which take roles in the context of the function of the Link. A point-to-point Link can be a TE Link and support parameters such as capacity, delay etc. These parameters depend on the type of technology that supports the link.

A Link can be terminated on two or more FDs. This provides support for technologies such as PON and Layer 2 MAC in MAC configurations.

The LinkPort further details the relationship between FD and Link for asymmetric cases.

A FD may aggregate Links (see section 4.2.5).

The Link can support multiple transport layers via the associated LTP object. An instance of Link can be formed with the necessary properties according to the degree of virtualization. For implementation optimization, multiple layer-specific links can be merged and represented as a single Link instance.

4.1.4. Network Element (NE)

The NetworkElement (NE) object class represents a network element (traditional NE) in the data plane or a virtual network element visible in an interface where virtualization is used.

In the direct interface from a SDN controller to a network element in the data plane, the NE object defines the scope of control for the resources within the network element, e.g., internal transfer of user information between the external terminations (ports), encapsulation, multiplexing/demultiplexing, and OAM functions, etc. The NE provides the scope of the naming space for identifying objects representing the resources within the network element.

Where virtualization is employed, the NE object represents a virtual NE (VNE). The mapping of the VNE to the NEs is the internal matter of the SDN controller that offers the view of the VNE. Via the interface between hierarchical SDN controllers, NE instances can be created (or deleted) for providing (or removing) virtual views of the combination of slices of network elements in the data plane.

4.2. Relationships between Object Classes of the Topology Subset

4.2.1. ForwardingDomain Recursive Aggregation (HigherLevelFdEncompassesLowerLevelFds Aggregation)

Figure 4-2 below provides a pictorial example of ForwardingDomain (FD) recursion with Links.

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
```

```
I              ForwardingDomain recursion with Links               I
I                                                                  I
I                                                                  I
I                                                                  I
I                                                                  I
I                     (only in PDF version)                        I
I                                                                  I
I                                                                  I
I                                                                  I
I                                                                  I
I                                                                  I
-------------------------------------------------------------------
```

Figure 4-2 ForwardingDomain recursion with Links

Figure 4-2 shows a UML fragment including the Link and ForwardingDomain (FD). For simplicity it is assumed here that the Links and FDs are for a single LayerProtocol (LP) although it can be seen from the detailed figure earlier in this section that both a FD and link can support a list of LPs.

The pictorial form shows a number of instances of FD interconnected by Links and shows nesting of FDs. The recursive aggregation "HigherLevelFdEncompassesLowerLevelFds" relationship (represented by an open diamond) supports the FD nesting but it should be noted that this is intentionally showing no lifecycle dependency between the lower FDs and the higher ones that nest them (to do this composition, a black diamond would have been used instead of the open diamond). This is to allow for rearrangements of the FD hierarchy (e.g. when regions of a network are split or merged). This emphasizes that the nesting is an abstraction rather than decomposition. The underlying network still operates regardless of how it is perceived in terms of aggregating FDs. The model allows for only one hierarchy.

4.2.2. Network Elements encompassing ForwardingDomains (NeEncompassesFds Aggregation)

Figure 4-3 below provides a pictorial example of ForwardingDomain (FD) recursion with Links and NEs.

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I          ForwardingDomain recursion with Links and NEs            I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                      (only in PDF version)                        I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   ---------------------------------------------------------------------
```

Figure 4-3 ForwardingDomain recursion with Links and NEs

Figure 4-3 above shows an overlay of NetworkElement (NE) on the ForwardingDomains and a corresponding fragment of UML showing only the ForwardingDomain and NetworkElement classes.

The figure emphasizes that one level of abstraction of ForwardingDomain is bounded by an NE. This is represented in the UML fragment by the composition association (black diamond) that explains that there is a lifecycle dependency in that the ForwardingDomain at this level that cannot exist without the NE. The figure also shows that a ForwardingDomain need not be bounded by an NE (as explained in the UML fragment by the 0..1 composition) and that a ForwardingDomain may have smaller scope than the whole NE (even when considering only a single LayerProtocol as described below).

In one of the cases depicted (e.g., the right hand side NE encompassing two FDs), the two ForwardingDomains in the NE are completely independent. In the other cases depicted (e.g., the left

hand side NE encompassing three FDs) the subordinate ForwardingDomains are themselves joined by Links emphasizing that the NE does not necessarily represent the lowest level of relevant network decomposition.

The figure also emphasizes that just because one ForwardingDomain at a particular level of decomposition of the network happens to be the one bounded by an NE does not mean that all ForwardingDomains at that level are also bounded by NEs.

4.2.3. ForwardingDomain association with LTPs (FdAggregatesLtps Composition)

An instance of FD is associated with zero or more LTP objects via the "FdAggregatesLtps" composition.

4.2.4. ForwardingDomain aggregating Links (FdEncompassesLinks)

A ForwardingDomain can aggregate links. An example of ForwardingDomain Recursive Aggregation with Links is shows in section 4.2.1 above.

However, the FdAggregatesLink association is not modeled because this association can be inferred from the higherLevelFdContainsLowerLevelFd association together with the linkHasAssociatedFds association.

4.2.5. ForwardingDomain aggregating NEs

A ForwardingDomain can aggregate Network Elements. An example of ForwardingDomain Recursive Aggregation with Links and NEs is shown in section 4.2.2 above.

However, the FdAggregatesNe association is not modeled because this association can be inferred from higherLevelFdContainsLowerLevelFd association and together with the NeEncompassesFd association.

5. Detailed Description of the Topology Subset

The two key classes related to Topology are the ForwardingDomain (FD) and the Link. For simple cases the FD represents the switching

capability in the network and the Link represents adjacency. These are depicted in the context of other model classes in Figure 5-1.

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I     Object Classes and Relationships in the Topology Subset       I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                      (only in PDF version)                        I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   ---------------------------------------------------------------------
```

Figure 5-1 Object Classes and Relationships in the Topology Subset

Figure 5-1 shows a lightweight view of the model omitting the attributes (where appropriate these will be described later in this section).

The FD and Link will be described in detail later in the document. Figure 5-1 focuses on interrelationships and these will be the focus of this section. The figure shows that:

- An FD may be a subordinate part of a NetworkElement (NE) or may be larger than, and independent of, any NE.

- An FD may encompass lower level FDs. This may be such that:

  o A FD directly contained in an NE is divided into smaller parts

  - o A FD not encompassed by an NE is divided into smaller parts some of which may be encompassed by NEs

  - o The FD represents the whole network

- An FD encompasses Links that interconnect any FDs encompassed by the FD

- A Link may aggregate Links in several ways

  - o In parallel where several links are considered as one

  - o In series where Links chain to form a Link of a greater span

    - . Note that this case requires further development in the model

- A Link has associated FDs that it interconnects

  - o A Link may interconnect 2 or more FDs

    - . Note that it is usual for a Link to interconnect 2 FDs but there are cases where many FDs may be interconnected by a Link

- A Link has LinkPorts that represent the ports of the Link itself

  - o LinkPorts are especially relevant for multi-ended asymmetric Link

- A LinkPort aggregates LogicalTerminationPoints (LTPs) that bound the Link. The LTP represent a stack LayerProtocol terminations where the details of each is held in the LayerProtocol (LP). The LTP may be:

  - o Part of an NE

  - o Conceptually independent from any NE

- A LinkPort references LTPs on which the Link associated to the LE terminates

Both the Link and FD are subclasses of ForwardingEntity (an abstract class, i.e. a class that will never be instantiated) and hence they can acquire contents from the conditional packages (_Pacs). The conditional packages provide all key topology properties.

5.1. Forwarding Entity

As noted in the previous section the two key topology classes are Forwarding Domain (FD) and Link (L).

The FD topological component is used to show the potential to enable forwarding. At the lowest level of recursion, an FD (within a network element (NE)) represents a switch matrix (e.g., a fabric). Note that an NE can encompass multiple switch matrices (FDs).

As noted earlier the Link models adjacency between two or more Forwarding Domains (FD).

Both the link and the FD have the potential to handle more than one layerProtocol (both have a layerProtocolNameList attribute).

As shown in Figure 5-1 an object class "ForwardingEntity" has been defined to collect topology-related properties (characteristics etc.) that are common for FD and Link.

A ForwardingEntity is an abstract representation of the emergent effect of the combined functioning of an arrangement of components (running hardware, software running on hardware, etc). The effect can be considered as the realization of the potential for apparent communication adjacency for entities that are bound to the terminations at the boundary of the ForwardingEntity.

The ForwardingEntity enables the creation of constrained forwarding to achieve the apparent adjacency. The apparent adjacency has intended performance degraded from perfect adjacency and a statement of that degradation is conveyed via the attributes of the packages associated with this class. In the model both ForwardingDomain and Link are ForwardingEntities.

This abstract class is used as a modeling approach to apply packages of attributes to both Link and ForwardingDomain. Link and ForwardingDomain are the key ForwardingEntities.

5.2. Characteristics of Topological Entity

As noted above the characteristic of a TopologicalEnity are covered by the conditional packages (_PACs).

```
   ----------------------------------------------------------------------
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I           Conditional Packages of Topological Entity               I
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I                     (only in PDF version)                          I
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I                                                                    I
   I                                                                    I
```

Figure 5-2 Conditional Packages of Topological Entity

5.2.1. Risk (RiskParameter_Pac)

The risk characteristics of a ForwardingEntity come directly from the underlying physical realization.

The risk characteristics propagate from the physical realization to the client and from the server layer to the client layer, this propagation may be modified by protection.

A ForwardingEntity may suffer degradation or failure as a result of a problem in a part of the underlying realization.

The realization can be partitioned into segments which have some relevant common failure modes.

There is a risk of failure/degradation of each segment of the underlying realization.

Each segment is a part of a larger physical/geographical unit that behaves as one with respect to failure (i.e. a failure will have a high probability of impacting the whole unit (e.g. all fibers in the same cable).

Disruptions to that larger physical/geographical unit will impact (cause failure/errors to) all ForwardingEntities that use any part of that larger physical/geographical entity.

Any ForwardingEntity that uses any part of that larger physical/geographical unit will suffer impact and hence each ForwardingEntity shares risk.

The identifier of each physical/geographical unit that is involved in the realization of each segment of a Topological entity can be listed in the RiskParameter_Pac of that ForwardingEntity.

A segment has one or more risk characteristic.

Shared risk between two ForwardingEntities compromises the integrity of any solution that use one of those ForwardingEntity as a backup for the other.

Where two ForwardingEntities have a common risk characteristic they have an elevated probability of failing simultaneously compared to two ForwardingEntities that do not share risk characteristics.

- riskCharacteristicList: A list of risk characteristics (RiskCharacteristic) for consideration in an analysis of shared risk. Each element of the list represents a specific risk consideration.

- RiskCharacteristic: The information for a particular risk characteristic where there is a list of risk identifiers related to that characteristic. It includes:

  - o riskCharacteristicName: The name of the risk characteristic. The characteristic may be related to a specific degree of closeness. For example a particular characteristic may apply to failures that are localized (e.g. to one side of a road) where as another characteristic may relate to failures that have a broader impact (e.g. both sides of a road that crosses a bridge). Depending upon the importance of the traffic being routed different risk characteristics will be evaluated.

  - o riskIdentifierList: A list of the identifiers of each physical/geographic unit (with the specific risk characteristic) that is related to a segment of the ForwardingEntity.

5.2.2. TransferCost_Pac

The cost characteristics of a ForwardingEntity not necessarily correlated to the cost of the underlying physical realization.

They may be quite specific to the individual ForwardingEntity e.g. opportunity cost. Relates to layer capacity

There may be many perspectives from which cost may be considered for a particular ForwardingEntity and hence many specifc costs and potentially cost algorithms.

Using an entity will incur a cost.

- costCharcteristicList: The list of costs (CostCharacteristic) where each cost relates to some aspect of the Link

  o CostCharcteristic: The information for a particular cost characteristic

    . costName: The cost characteristic will related to some aspect of the ForwardingEntity (e.g. $ cost, routing weight). This aspect will be conveyed by the costName

    . costValue: The specific cost.

    . costAlgorithm: The cost may vary based upon some properties of the ForwardingEntity. The rules for the variation are conveyed by the costAlgorithm.

5.2.3. TransferTiming_Pac

A link will suffer effects from the underlying physical realization related to the timing of the information passed by the link.

- fixedLatencyCharacteristic: A ForwardingEntity suffers delay caused by the realization of the servers (e.g. distance related; FEC encoding etc.) along with some client specific processing. This is the total average latency effect of the ForwardingEntity

- jitterCharacteristic: High frequency deviation from true periodicity of a signal and therefore a small high rate of change of transfer latency. Applies to TDM systems (i.e., not packet based systems).

- wanderCharacteristics: Low frequency deviation from true periodicity of a signal and therefore a small low rate of

change of transfer latency. Applies to TDM systems (i.e., not packet based systems).

- queuingLatencyList: The effect on the latency of a queuing process. This only has significant effect for packet based systems and has a complex characteristic (QueuingLatency).

  o QueuingLatency: Provides information on latency characteristic for a particular stated trafficProperty.

5.2.4. TransferIntegrity_Pac

Transfer integrity characteristic covers expected (specified) error, loss and duplication signal content as well as any damage of any form to total link and to the client signals.

- errorCharacteristic: describes the degree to which the signal propagated can be errored. Applies to TDM systems as the errored signal will be propagated and not packet as errored packets will be discarded.

- lossCharacteristic: Describes the acceptable characteristic of lost packets where loss may result from discard due to errors or overflow. Applies to packet systems and not TDM (as for TDM errored signals are propagated unless grossly errored and overflow/underflow turns into timing slips).

- repeatDeliveryCharacteristic: Primarily applies to packet systems where a packet may be delivered more than once (in fault recovery for example). It can also apply to TDM where several frames may be received twice due to switching in a system with a large differential propagation delay.

- deliveryOrderCharacteristic: Describes the degree to which packets will be delivered out of sequence. Does not apply to TDM as the TDM protocols maintain strict order.

- unavailableTimeCharacteristic: Describes the duration for which there may be no valid signal propagated.

- serverIntegrityProcessCharacteristic: Describes the effect of any server integrity enhancement process on the characteristics of the ForwardingEntity.

5.2.5. TransferCapcity_Pac

The ForwardingEntity derives capacity from the underlying realization.

A ForwardingEntity may be an abstraction and virtualization of a subset of the underlying capability offered in a view or may be directly reflecting the underlying realization.

A ForwardingEntity may be directly used in the view or may be assigned to another view for use.

The clients supported by a multi-layer ForwardingEntity may interact such that the resources used by one client may impact those available to another. This is derived from the LTP spec details.

A ForwardingEntity represents the capacity available to user (client) along with client interaction and usage.

A ForwardingEntity may reflect one or more client protocols and one or more members for each profile.

- totalPotentialCapacity: A "best case" view of the capacity of the ForwardingEntity assuming that any shared capacity is available to be taken.

Note that this area is still under development to cover concepts such as:

- exclusiveCapacityList: The capacity allocated to this ForwardingEntity for its exclusive use

- sharedCapacityList: The capacity allocated to this ForwardingEntity that is not exclusively available as it is shared with others.

- assignedAsExclusiveCapacityList: The capacity assigned from this TopologicalEnity to another ForwardingEntity for its exclusive use

- assignedAsSharedCapacityList: The capacity assigned to one or more other ForwardingEntities for shared use where the interaction follows some stated algorithm.

- Capacity which includes:

  - totalSize

  - numberOfUsageInstances

  - maximumUsageSize

  - numberingRange

5.2.6. Validation_Pac

Validation covers the various adjacenct discovery and reachability verification protocols. Also may cover Information source and degree of integrity.

- validationMechanismList: Provides details of the specific validation mechanism(s) used to confirm the presence of an intended ForwardingEntity.

5.2.7. LayerProtocolTransition_Pac

Relevant for a Link that is formed by abstracting one or more LTPs (in a stack) to focus on the flow and deemphasize the protocol transformation.

This abstraction is relevant when considering multi-layer routing.

The layer protocols of the LTP and the order of their application to the signal is still relevant and need to be accounted for. This is derived from the LTP spec details.

This Pac provides the relevant abstractions of the LTPs and provides the necessary association to the LTPs involved.

Links that included details in this Pac are often referred to as Transitional Links.

- transitionedLayerProtocolList: Provides the ordered structure of layer protocol transitions encapsulated in the ForwardingEntity. The ordering relates to the LinkEnd role.

6. Purpose Specific IM Example - Transport API Topology Service

In order to provide some further clarity, this section provides a high level introduction to a Purpose Specific IM, the Transport API (T-API) Topology service, which has been derived from the ONF Common Information Model (ONF-CIM) according to the principles in [I-D.betts].

The context of the T-API refers to the scope and control and naming that a particular SDN controller, manager or a client application has with respect to the information it operates on internally or exchanges over an interface. The following sections further describe this purpose specific IM and relationship to the ONF-CIM.

6.1. T-API IM Constructs

The T-API IM uses terminology that is considered to be more familiar to the transport network management community and maps to the constructs defined in the ONF-CIM CNM Topology model. The following table provides a high level summary of the mapping of the constructs relevant to the T-API Topology Service.

Mapping of CIM and T-API IM Constructs

ONF-CIM CNM Terminology	T-API IM Terminology
NetworkControlDomain	Context
	Topology
ForwardingDomain (FD)	Node
	Link
Link	TransitionalLink
	NodeEdgePoint
LogicalTerminationPoint (LTP)	ServideEndPoint

The following provides a brief description of these T-API IM constructs.

- o Link: A Link is an abstract representation of the effective adjacency between two or more associated Nodes in a Topology. It is terminated by Node-Edge-Points of the associated Nodes.

- o Node: A Node is an abstract representation of the forwarding-capabilities of a particular set of Network Resources. It is described in terms of an aggregation of set of ports (Node-Edge-Point) belonging to those Network Resources and the potential to enable forwarding of information between those edge ports.

- o Node-Edge-Point: A Node-Edge-Point represents the inward network-facing aspects of the edge-port functions that access the forwarding capabilities provided by the Node. Hence it provides an encapsulation of addressing, mapping, termination, adaptation and OAM functions of one or more transport layers (including circuit and packet forms) performed at the entry and exit points of the Node.

- o Topology: A Topology is an abstract representation of the topological-aspects of a particular set of Network Resources. It is described in terms of a network of set of Nodes and Links that enable the forwarding-capabilities of that particular set of Network Resources.

- o Service-End-Point: A Service-End-Point represents the outward customer-facing aspects of the edge-port functions that access the forwarding capabilities provided by the Node. Hence it provides a limited, simplified view of interest to external clients (e.g. shared addressing, capacity, resource availability, etc) that enable the clients to request connectivity without the need to understand the provider network internals.

- o Transitional Link: A topological component that consists of the link port at the edge of one node and a corresponding link port at the edge of another node that operates on different layers or whose layer is the same but with different Layer

Information. A transitional link is supported/implemented by transport processing functions (e.g., adaptation/termination). A transitional link can be partitioned into parallel transitional links, or a concatenation of transitional links. It can also be partitioned into a concatenation of transitional links and zero or more links.

6.2. T-API Topology Service IM

The resultant high-level description for the T-API Topology Service constructs, based upon the pruned and refactored ONF-CIM, and the related Topology Service APIs are provided in Figure 6-1 below.

```
   ---------------------------------------------------------------------
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                     Topology Service Skeleton                       I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                      (only in PDF version)                        I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   I                                                                   I
   ---------------------------------------------------------------------
```

Figure 6-1 Topology Service Skeleton

The T-API Topology Service API enables the API client to, for example, retrieve Topology, Node, Link, and Edge-Point details.

- o Topology details: returns attributes of the Topology identified by the provided input ID. This includes references to lower-level Nodes and Links encompassed by that Topology. A NULL input value is expected to return the top-most Topology that corresponds to the scope of the entire Context including any Off-Network-Links.

- o Node details: Returns attributes of the Node identified by the provided input ID. Includes references to Node-Edge-Points aggregated by the Node, and attributes representing the identification, naming, states and forwarding capabilities of the Node.

- o Link details: Returns attributes of the Link identified by the provided input ID. Includes references to Node-Edge-Points terminating the Link, and references to the Nodes associated by the Link.

- o Node-Edge-Point details: Returns attributes of the Node-Edge-Point identified by the provided input ID, including references to Service-End-Points mapped to this Node-Edge-Point.

The API supports a retrieve-scope filter: LayerProtocol list. If set, the API call will return output that is relevant to the specified Layer only.

7. Usage of the IM Topology Subset regarding TE Topology DM

As discussed earlier, a data model (DM) may be derived from an IM. Examples of YANG DMs derived according to automated translation tools based upon mapping guidelines are provided in [OSSDN SNOMASS] at https://github.com/OpenNetworkingFoundation/Snowmass-ONFOpenTransport/tree/develop/YANG. It is possible to leverage the IM

Topology Subset to assess the consistency and completeness of related YANG modules under development.

8. Security Considerations

This informational document is intended only to provide a description of an interface-protocol-neutral information model, and the security concerns are therefore out of the scope of this document.

9. IANA Considerations

This document includes no request to IANA.

10. Conclusions

The information modeling described in this draft, which is relevant to Network Topology [ONF TR-512] [OSSDN SNOWMASS], can be leveraged in assessing the consistency and completeness of related YANG modules under development.

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.

11.2. Informative References

[I-D.betts] Betts, M., Davis, N., Lam, K., Zeuner, B., Mansfield, S. and P. Doolan, "Framework for Deriving Interface Data Schema from UML Information Models", draft-betts-netmod-framework-data-schema-uml-04 (work in progress), October 2016

[I-D.mansfield] Mansfield, S., Zeuner, B., Davis, N., Yun, X., Tochio, Y., Lam, K. and E. Varma, "Guidelines for Translation of UML Information Model to YANG Data Model", draft-mansfield-netmod-uml-to-yang-03 (work in progress), October 2016

[ONF TR-512] ONF TR-512 "ONF-CIM Core Model base document 1.2" (https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-512_CIM_(CoreModel)_1.2.zip), September 2016

[ONF TR-513] ONF TR-513 " Common Information Model Overview 1.2" (https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-513_CIM_Overview_1.2.pdf), September 2016

[ONF TR-514] ONF TR-514 "UML Modeling Guidelines 1.2" (https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-514_UML_Modeling_Guidelines_v1.2.pdf), September 2016

[ONF TR-515] ONF TR-515 "Papyrus Guidelines 1.2" (https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-515_Papyrus_Guidelines_v1.2.pdf), September 2016

[OSSDN SNOWMASS] Open Source SDN SNOWMASS/Open Transport API Specifications (https://github.com/OpenNetworkingFoundation/Snowmass-ONFOpenTransport)

[G.7711] Recommendation ITU-T G.7711/Y.17022 "Generic protocol-neutral information model for transport resources", September 2016

[G.874.1] Recommendation ITU-T G.874.1 "Optical transport network: Protocol-neutral management information model for the network element view", September 2016

[G.8052] Recommendation ITU-T G.8052/Y.1346 "Protocol-neutral management information model for the Ethernet Transport capable network element", September 2016

[G.8152] Recommendation ITU-T G.8152/Y.1375 "Protocol-neutral management information model for the MPLS-TP network element", September 2016

[G.852.2] Recommendation ITU-T G.852.2 "Enterprise viewpoint description of transport network resource model", March 1999

[TMF612] TM Forum 612 "MTOSI Information Agreement", October 2014

12. Contributors

Karthik Sethuraman
NEC

Email: karthik.sethuraman@necam.com

13. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Kam Lam
Alcatel-Lucent, USA

Phone: +1 732 331 3476
Email: kam.lam@alcatel-lucent.com

Eve Varma
Alcatel-Lucent, USA

Email: eve.varma@alcatel-lucent.com

Paul Doolan
Coriant, Germany

Phone: +1 972 357 5822
Email: paul.doolan@coriant.com

Malcolm Betts
ZTE, China

Phone: +1 678 534 2542
Email: malcolm.betts@zte.com.cn

Nigel Davis
Ciena, UK

Email: ndavis@ciena.com

Bernd Zeuner
Deutsche Telekom, Germany

Phone: +49 6151 58 12086
Email: b.zeuner@telekom.de

Italo Busi
Huawei, China

Email: Italo.Busi@huawei.com

Scott Mansfield
Ericsson, Sweden

Phone: 1 724 931 9316
Email: scott.mansfield@ericsson.com

Yuji Tochio
Fujitsu, Japan

Phone: 81 44 754 8829
Email: tochio@jp.fujitsu.com

Ricard Vilalta
CTTC, Spain

Phone:
Email: ricard.vilalta@cttc.es

Victor Lopez
Telefonica, Spain

Phone:
Email: victor.lopezalvarez@telefonica.com

Network Working Group

Internet Draft

Intended status: Informational

Expires: January 2016

Young Lee (Editor)
Dhruv Dhody
Huawei

Sergio Belotti
Alcatel-Lucent

Khuzema Pithewan
Infinera

Daniele Ceccarelli
Ericsson

July 27, 2015

Requirements for Abstraction and Control of Transport Networks

draft-lee-teas-actn-requirements-01.txt

Abstract

This draft provides a set of requirements for abstraction and control of transport networks.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This draft provides a set of requirements for ACTN identified in various use-cases of ACTN. [ACTN-frame] defines the base reference architecture and terminology. [ACTN-PS] provides problem statement and gap analysis.

Section 2 provides high-level ACTN requirements. Sections 3-5 provide the list of ACTN use-cases and the detailed requirement analysis of these use-cases.

2. High-level ACTN requirements

1. Requirement 1: Single Virtualized Network Topology

   Ability to build virtual network operation infrastructure based on multi-layer, multi-domain topology abstracted from multiple physical network controllers (e.g., GMPLS, OpenFlow, PCE, NMS, etc.)

   Reference: [KLEE], [LOPEZ], [DHODY], [CHENG].

2. Requirement 2: Policy Enforcement

   Ability to provide service requirement/policy (Between Customer and Network) and mechanism to enforce service level agreement.

   - Endpoint selection policy, routing policy, time-related policy, etc.

   Reference: [KLEE], [LOPEZ], [SHIN], [DHODY], [FANG].

3. Requirement 3: VN Query

   Ability to request/respond VN Query (Can you give me VN(s)?)

   - Request Input:

     - VN end-points (CE end)

     - VN Topology Service-specific Multi-Cost Objective Function

     - VN Topology diversity (e.g., VN1 and VN2 must be disjoint)

     - VN Topology type: path, graph

- Response includes VN topology
    - Exact
    - Potential

Reference: [KUMAKI], [FANG], [CHENG].

4. Requirement 4: VN Instantiate

Ability to request/confirm VN Instantiation

- VN instance ID
- VN end-points
- VN constraints requirement
    - Latency only, B/W guarantee, Latency and B/W guarantee together
- VN diversity
    - Node/Link disjoint from other VNs
- VN level diversity (e.g., VN1 and VN2 must be disjoint)
- VN type
    - Path (tunnel), Node/Links (graph)
- VN instance ID per service (unique id to identify VNs)

Reference: [KUMAKI], [FANG], [CHENG].

5. Requirement 5: Dynamic VN Control

Dynamic/On-demand VN Modification/Confirmation with feedback loop to the customer

- Traffic monitoring and control policies sent to the network
- Network states based traffic optimization policies

- Utilization Monitoring (Frequency of report)

- Abstraction of Resource Topology reflecting these service-related parameters

Reference: [XU], [XU2], [DHODY], [CHENG].

6. Requirement 6: VN Lifecycle M&O

VN lifecycle management/operation

- Instantiate

- Delete

- Modify

- Update (VN level OAM Monitoring) under policy agreement

Reference: [FANG], [KUMAKI], [LOPEZ].

7. Requirement 7: VN Service Operation

Ability to setup and manage end-2-end service on the VN involving multi-domain, multi-layer, meeting constraints based on SLAs.

Reference: [LOPEZ], [KUMAKI], [CHENG], [DHODY], [FANG], [KLEE].

8. Requirement 8: Multi-destination Coordination

Coordination of multi-destination service requirement/policy to support dynamic applications such as VM migration, disaster recovery, load balancing, etc.

- Service-policy primitives and its parameters

Reference: [FANG], [LOPEZ], [SHIN].

9. Requirement 9: Multi-domain & Multi-layer Coordination

Ability to Coordinate multi-domain and multi-layer path computation and setup operation (network)

- Computes E2E path across multi-domain (based on abstract topology from each domain)

- Determines the domain sequence

- Request path signaling to each domain controller

- Find alternative path if any of the domain controllers cannot find its domain path

Reference: [CHENG], [DHODY], [KLEE], [LOPEZ], [SHIN], [SUZUKI].

10. Requirement 10: E2E Path Restoration

Ability to perform E2E Path Restoration Operation

- Intra-domain recovery

- Cross-domain recovery

Reference: [CHENG], [KLEE], [DHODY], [LOPEZ], [SHIN].

11. Requirement 11: Dynamicity of network control operations

The ACTN interfaces should support dynamicity nature of network control operations. This includes but not limited to the following:

- Real-time VN control (e.g., a fast recovery/reroute upon network failure).
- Fast convergence of abstracted topologies upon changes due to failure or reconfiguration across the network domain view, the multi-domain network view and the customer view.
- Large-scale VN operation (e.g., ability to query tens of thousands of nodes and connectivity) for time-sensitive applications.

Reference: [SHIN], [XU], [XU2], [KLEE], [KUMAKI], [SUZUKI].

12. Requirement 12: VN confidentiality/security

- A VN customer MUST not control other customer’s virtual network
- A VN customer MUST not see any routing information (e.g. IGP database, TE database) on other customer’s virtual network

Reference: [KUMAKI], [FANG], [LOPEZ]

3. ACTN Use-Cases

Listed below is a set of high-level requirements identified by each of the ACTN use-cases:

- [CHENG] (ACTN Use-cases for Packet Transport Networks in Mobile Backhaul Networks)

  - o Faster End-to-End Enterprise Services Provisioning
  - o Multi-layer coordination in L2/L3 Packet Transport Networks
  - o Optimizing the network resources utilization (supporting various performances monitoring matrix, such as traffic flow statistics, packet delay, delay variation, throughput and packet-loss rate)
  - o Virtual Networks Operations for multi-domain Packet Transport Networks

- [DHODY] (Packet Optical Integration (POI) Use Cases for Abstraction and Control of Transport Networks (ACTN))

  - o Packet Optical Integration to support Traffic Planning, performance Monitoring, automated congestion management and Automatic Network Adjustments
  - o Protection and Restoration Synergy in Packet Optical Multi-layer network.
  - o Service Awareness and Coordination between Multiple Network Domains

- [FANG] (ACTN Use Case for Multi-domain Data Center Interconnect)

  - o Multi-domain Data Center Interconnection to support VM Migration, Global Load Balancing, Disaster Recovery, On-demand Virtual Connection/Circuit Services
  - o The interfaces between the Data Center Operation and each transport network domain SHOULD support standards-based

abstraction with a common information/data model to support the following:

. Network Query (Pull Model) from the Data Center Operation to each transport network domain to collect potential resource availability (e.g., BW availability, latency range, etc.) between a few data center locations.

. Network Path Computation Request from the Data Center Operation to each transport network domain to estimate the path availability.

. Network Virtual Connections/Circuits Request from the Data Center Operation to each transport domain to establish end-to-end virtual connections/circuits (with type, concurrency, duration, SLA.QoS parameters, protection.reroute policy options, policy constraints such as peering preference, etc.).

. Network Virtual Connections/Circuits Modification Request

- [KLEE] (ACTN Use-case for On-demand E2E Connectivity Services in Multiple Vendor Domain Transport Networks)

o Two-stage path computation capability in a hierarchical control architecture (MDSC-PNC) and a hierarchical composition of integrated network views

o Coordination of signal flow for E2E connections and management.

o Abstraction of:

. Inter-connection data between domains

. Customer Endpoint data

. The multiple levels/granularities of the abstraction of network resource (which is subject to policy and service need).

. Any physical network constraints (such as SRLG, link distance, etc.) should be reflected in abstraction.

. Domain preference and local policy (such as preferred peering point(s), preferred route, etc.), Domain network capability (e.g., support of push/pull model).

- [KUMAKI] (ACTN : Use case for Multi Tenant VNO)

o On-demand Virtual Network Service Creation
o Domain Control Plane/Routing Layer Separation
o Independent service Operation for Virtual Services from control of other domains
o Multiple service level support for each VN (e.g., bandwidth and latency for each VN service).
o VN diversity/survivability should be met in physical network mapping.
o VN confidentiality and sharing constraint should be supported.

- [LOPEZ] (ACTN Use-case for Virtual Network Operation for Multiple Domains in a Single Operator Network)

  o Creation of a global abstraction of network topology: The VNO Coordinator assembles each domain level abstraction of network topology into a global abstraction of the end-to-endnetwork.
  o End-to-end connection lifecycle management
  o Invocation of path provisioning request to each domain (including optimization requests)
  o Invocation of path protection/reroute to the affected domain(s)
  o End-to-end network monitoring and fault management. This could imply potential KPIs and alarm correlation capabilities.
  o End-to-end accounting and generation of detailed records for resource usage
  o End-to-end policy enforcement

- [SHIN] (ACTN Use-case for Mobile Virtual Network Operation for Multiple Domains in a Single Operator Network)

  o Resource abstraction: operational mechanisms in mobile backhaul network to give the current network usage information for dynamic and elastic applications be provisioned dynamically with QoS guarantee.

  o Load balancing or for recovery, the selection of core DC location from edge constitutes a data center selection problem.

  o Multi-layer routing and optimization, coordination between these two layers.

- [SUZUKI] (Use-case and Requirements for Multi-domain Operation Plane Change)

o Operational state data synchronization between multi-domain controllers

- [XU] (Use Cases and Requirements of Dynamic Service Control based on Performance Monitoring in ACTN Architecture)

  o Dynamic Service Control Policy enforcement and Traffic/SLA Monitoring:
    . Customer service performance monitoring strategy, including the traffic monitoring object (the service need to be monitored)
    . monitoring parameters (e.g., transmitted and received bytes per unit time),
    . traffic monitoring cycle (e.g., 15 minutes, 24 hours),
    . threshold of traffic monitoring (e.g., high and low threshold), etc.

- [XU2] (Requirements of Abstract Alarm Report in ACTN architecture

  o Dynamic abstract alarm report

3.1. Two categories of requirements

This section provides a summary of use-cases in terms of two categories: (i) service-specific requirements; (ii) network-related requirements.

Service-specific requirements listed below are uniquely applied to the work scope of ACTN. Service-specific requirements are related to virtual service coordination function defined in Section 3. These requirements are related to customer’s VNs in terms of service policy associated with VNs such as service performance objectives, VN endpoint location information for certain required service-specific functions (e.g., security and others), VN survivability requirement, or dynamic service control policy, etc.

Network-related requirements are related to virtual network operation function defined in Section 3. These requirements are related to multi-domain and multi-layer signaling, routing, protection/restoration and synergy, re-optimization/re-grooming, etc. These requirements are not inherently unique for the scope of ACTN but some of these requirements are in scope of ACTN, especially

for coherent/seamless operation aspect of multiple controller hierarchy.

The following table gives an overview of service-specific requirements and network-related requirements respectively for each ACTN use-case and identifies the work in scope of ACTN.

Use-case	Service-specific Requirements	Network-related Requirements	Control Functions/Data Models to be supported
[CHENG]	- E2E service provisioning - Performance monitoring - Resource utilization abstraction	- Multi-layer (L2/L2.5) coordination - VNO for multi-domain transport networks	- Dynamic multi-layer coordination function based on utilization - YANG for utilization abstraction
[DHODY]	- Service awareness/ coordination between P/O.	- POI Performance monitoring - Protection/ Restoration synergy	- Customer's VN survivability policy enforcement for protection/restoration - YANG for Performance Monitoring
[FANG]	- Dynamic VM migration (service), Global load balancing (utilization efficiency), Disaster recovery - Service-aware network query - Service Policy Enforcement	- On-demand virtual circuit request - Network Path Connection request	- Multi-destination service selection policy enforcement function - YANG for Service-aware policy enforcement
[KLEE]		- Two stage path	- Multi-domain

		computation E2E signaling coordination - Abstraction of inter-domain info - Enforcement of network policy (peering, domain preference) - Network capability exchange (pull/push, abstraction level, etc.) - on-demand and long-lived end-to-end service provisioning and monitoring	service policy coordination to network primitives - YANG for Abstraction of peering/ boundary data
[KUMAKI]	- On-demand VN creation - Multi-service level for VN - VN survivability /diversity/confidentiality		- Dynamic VN creation, survivability with security/ confi-dentiality
[LOPEZ]	- E2E accounting and resource usage data - E2E service policy enforcement	- E2E connection management, path provisioning - E2E network monitoring and fault management	- Escalation of performance and fault management data to CNC and the policy enforcement - YANG for performance and fault management

[SHIN]	- Current network resource abstraction Endpoint/DC dynamic selection (for VM migration)	- LB for recovery - Multi-layer routing and optimization coordination	- Multi-layer routing and optimization - VN's dynamic endpoint selection policy.
[SUZUKI]		- Operational Data/State between multi-domain controllers	- Operations DB sync function across controllers - YANG for operational data/state model
[XU]/ [XU2]	- Dynamic service control policy enforcement - Dynamic service control	- Traffic monitoring - SLA monitoring	- Dynamic service control policy enforcement control - YANG for traffic monitoring abstraction, alarm abstraction.

4. ACTN interfaces requirements

This section provides detailed ACTN interface requirements for the two interfaces that are within the ACTN scope based on [ACTN-Frame] and the use-cases referenced in this document.

. CMI: CNC-MDSC Interface
. MPI: MDSC-PNC Interface

4.1. CMI Interface Requirements

Requirement
1. Security/Policy Negotiation (Who are you?) (Between CNC and MDSC)
- Configured vs. Discovered
- Trust domain verification (External Entity vs. Internal Service Department)
- Push/Pull support (for policy)
2. VN Topology Query (Can you give me VN?) (From CNC to MDSC)
- VN end-points (CE end)
- VN Topology Service-specific Multi-Cost Objective Function
    o Latency Map
    o Available B/W Map
    o Latency Map and Available B/W Map together
    o Other types
- VN Topology diversity
    o Node/Link disjoint from other VNs
    o VN Topology level diversity (e.g., VN1 and VN2 must be disjoint)
- VN Topology type
    o Path vector (tunnel)
    o Node/Links (graph)
3. VN Topology Query Response (From MDSC to CNC: Here’s the VN Topology that can be given to you if you accept)
- For VN Topology,
    o This is what can be reserved for you
    o This is what is available beyond what is given to you (potential)
4. VN Topology Abstraction Model (generic network model)
5. VN Topology Abstraction Model (Service-specific model that include customer endpoints)
6. Basic VN Instantiation Request/Confirmation (Between CNC and MDSC: I need VN for my service, please instantiate my VN)
- VN instance ID
- VN end-points
- VN service requirement
    o Latency only
    o B/W guarantee
    o Latency and B/W guarantee together
- VN diversity
    o Node/Link disjoint from other VNs
- VN level diversity (e.g., VN1 and VN2 must be disjoint)
- VN type
    o Path vector (tunnel)
    o Node/Links (graph)
- VN instance ID per service (unique id to identify VNs)
- If failed to instantiate the requested VN, say why

7. Dynamic/On-demand VN Instantiation/Modification and Confirmation with feedback loop (This is to be differentiated from Basic VN Instantiation)
- Performance/Fault Monitoring
- Utilization Monitoring (Frequency of report)
- Abstraction of Resource Topology reflecting these service-related parameters
- Dynamic Policy enforcement

8. VN lifecycle management/operation
- Create (same as VN instantiate Request)
- Delete
- Modify
- Update (VN level OAM Monitoring) under policy agreement
9. Coordination of multi-destination service requirement/policy to support dynamic applications such as VM migration, disaster recovery, load balancing, etc.
- Service-policy primitives and its parameters

4.2. MPI (MDSC-PNC Interface)

Requirement
1. Security/Policy negotiation (who are you?)
- Exchange of key, etc.
- Domain preference + local policy exchange
- Push/Pull support
- Preferred peering points
- Preferred route
- Reroute policy
- End-point mobility (for multi-destination)
2. Topology Query /Response (Pull Model from MDSC to PNC: Please give me your domain topology)
- TED Abstraction level negotiation
- Abstract topology (per policy)
    o Node/Link metrics
    o Node/Link Type (Border/Gateway, etc.)
    o All TE metrics (SRLG, etc.)
    o Topology Metrics (latency, B/W available, etc.)

3. Topology Update (Push Model from PNC to MDSC)
- Under policy agreement, topology changes to be pushed to MDSC from PNC

4. VN Path Computation Request (From MDSC to PNC: Please give me a path in your domain)
- VN Instance ID (Note: this is passed from CNC to MDSC)
- End-point information
- CE ends
- Border points (if applicable)
- All other PCE request info (PCEP)

5. VN Path Computation Reply (here’s the path info per your request)
- Path level abstraction
- LSP DB
- LSP ID ??
- VN ID

6. Coordination of multi-domain Centralized Signaling (MSDC operation) Path Setup Operation
- MSDC computes E2E path across multi-domain (based on abstract topology from each PNC)
- MDSC determines the domain sequence
- MDSC request path signaling to each PNC (domain)
- MDSC finds alternative path if any of the PNCs cannot find its domain path
    o PNC will crankback to MDSC if it cannot find its domain

path
o PNC will confirm to MDSC if it finds its domain path

7. Path Restoration Operation (after an E2E path is setup successfully, some domain had a failure that cannot be restored by the PNC domain)
- The problem PNC will send this notification with changed abstract topology (computed after resource changes due to failure/other factors)
- MDSC will find an alternate E2E path based on the changes reported from PNC. It will need to update the E2E abstract topology and the affected CN’s VN topology in real-time (This refers to dynamic synchronization of topology from Physical topology to abstract topology to VN topology)
- MDSC will perform the path restoration signaling to the affected PNCs.

8. Coordination of Multi-destination service restoration operation (CNC have, for example, multiple endpoints where the source endpoint can send its data to either one of the endpoints)
- When PNC reports domain problem that cannot be resolved at MDSC level because of there is no network restoration path to a given destination.
- Then MDSC has Customers’ profile in which to find the customer has "multi-destination" application.
- Under policy A, MDSC will be allowed to reroute the customer traffic to one of the pre-negotiated destinations and proceed with restoration of this particular customer’s traffic.
- Under policy B, CNC may reroute on its VN topology level and push this to MDSC and MDSC maps this into its abstract topology and proceed with restoration of this customer’s traffic.
- In either case, the MDSC will proceed its restoration operation (as explained in Req. 6) to the corresponding PNCs.

9. MDSC-PNC policy negotiation is also needed as to how restoration is done across MDSC and PNCs.

10. Generic Abstract Topology Update per changes due to new path setup/connection failure/degradation/restoration
11. Service-specific Abstract Topology Update per changes due to new path setup/connection failure/degradation/restoration
12. Abstraction model of technology-specific topology element

5. References

5.1. Informative References

[ACTN-Frame] D. Ceccarelli, et al., "Framework for Abstraction and Control of Transport Networks", draft-ceccarelli-teas-actn-framework, work in progress.

[CHENG] W. Cheng, et. al., "ACTN Use-cases for Packet Transport Networks in Mobile Backhaul Networks", draft-cheng-actn-ptn-requirements, work in progress.

[DHODY] D. Dhody, et. al., "Packet Optical Integration (POI) Use Cases for Abstraction and Control of Transport Networks (ACTN)", draft-dhody-actn-poi-use-case, work in progress.

[FANG] L. Fang, "ACTN Use Case for Multi-domain Data Center Interconnect", draft-fang-actn-multidomain-dci, work in progress.

[KLEE] K. Lee, H. Lee, R. Vilata, V. Lopez, "ACTN Use-case for E2E Network Services in Multiple Vendor Domain Transport Networks", draft-klee-teas-actn-connectivity-multi-domain, work-in-progress.

[KUMAKI] K. Kumaki, T. Miyasaka, "ACTN : Use case for Multi Tenant VNO ", draft-kumaki-teas-actn-multitenant-vno, work in progress.

[LOPEZ] D. Lopez (Ed), "ACTN Use-case for Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-lopez-actn-vno-multidomains, work in progress.

[SHIN] J. Shin, R. Hwang, J. Lee, "ACTN Use-case for Mobile Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-shin-actn-mvno-multi-domain, work in progress.

[XU] Y. Xu, et. al., "Use Cases and Requirements of Dynamic Service Control based on Performance Monitoring in ACTN Architecture", draft-xu-actn-perf-dynamic-service-control, work in progress.

[XU2] Y. Xu, et. al., "Requirements of Abstract Alarm Report in ACTN architecture", draft-xu-teas-actn-abstract-alarm-report, work-in-progress.

[SUZUKI] T. Suzuki, et. al., "Use-case and Requirements for Multi-domain Operation Plane Change", draft-suzuki-teas-actn-multidomain-opc, work-in-progress.

6. Contributors

Contributors' Addresses

Kwangkook Lee
KT
Email: kwangkooglee@gmail.com

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

Yunbin Xu
CATR
Email: xuyunbin@mail.ritt.com.cn

Toshiaki Suzuki
Hitachi
Email: toshiaki.suzuki.cs@hitachi.com

Authors' Addresses

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

Dhruv Dhody
Huawei Technologies
Email: dhruv.ietf@gmail.com

Sergio Belotti
Alcatel Lucent
Via Trento, 30
Vimercate, Italy
Email: sergio.belotti@alcatel-lucent.com

Khuzema Pithewan
Infinera
Email: kpithewan@infinera.com

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

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 6, 2016

J. George
Google
L. Fang
Microsoft
E. Osborne
Level 3
R. Shakir
BT

July 5, 2015

MPLS / TE Model for Service Provider Networks
draft-openconfig-mpls-consolidated-model-01

Abstract

This document defines a framework for a YANG data model for configuring and managing label switched paths, including the signaling protocols, traffic engineering, and operational aspects based on carrier and content provider operational requirements.

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 6, 2016.

Table of Contents

1. Introduction

This document describes a YANG [RFC6020] data model for MPLS and traffic engineering, covering label switched path (LSP) configuration, as well as signaling protocol configuration. The model is intended to be vendor-neutral, in order to allow operators to manage MPLS in heterogeneous environments with physical or virtual devices (routers, switches, servers, etc.) supplied by multiple vendors. The model is also intended to be readily mapped to existing implementations, to facilitate support from as large a set of routing hardware and software vendors as possible.

1.1. Goals and approach

The focus area of the model in this revision, is to set forth a framework for MPLS, with hooks into which information specific to various signaling-protocols can be added. The framework is built around functionality from a network operator perspective rather than a signaling protocol-centric approach. For example, a traffic-

engineered LSP will have configuration relating to its path computation method, regardless of whether it is signaled with RSVP-TE or with segment routing. Thus, rather than creating separate per-signaling protocol models and trying to stitch them under a common umbrella, this framework focuses on functionality, and adds signaling protocol-specific information under it where applicable.

This model does not aim to be feature complete (i.e., cover all possible aspects or features of MPLS). Rather its development is driven by examination of actual production configurations in use across a number of operator network deployments.

Configuration items that are deemed to be widely available in existing major implementations are included in the model. Those configuration items that are only available from a single implementation are omitted from the model with the expectation they will be available in companion modules that augment the current model. This allows clarity in identifying data that is part of the vendor-neutral model.

An important aspect of the model is the representation of operational state data. This draft takes the approach described in [I-D.openconfig-netmod-opstate] and models configuration and operational state together. Thus, rather than building a separate tree of operational state, the operational state and configuration data are located in parallel containers at the leaves of the data model. This approach allows easy reuse of groupings across models, as well as making it easier to correlate configuration and state.

The consolidated MPLS model encompasses the signaling protocols, label-switched paths (configuration and operational state), and generic TE attributes. The model is designed from an operational and functional perspective, rather than focusing on protocol-centric configuration. This allows protocol-independent functions to be logically separated from protocol-specific details.

One question that arises in this approach is how the consolidated model is integrated with routing instances (e.g., VRFs). This model should be considered as part of a higher level network device model which includes definitions for other routing protocols and system services. For example, in [I-D.openconfig-netmod-model-structure], VRFs and other logical instances are defined with MPLS/TE components within VRFs as appropriate. In particular, some parts of the MPLS model would be instantiated within a VRF, while other parts would have common definitions across VRFs.

Where possible, naming in the model follows conventions used in available standards documents, and otherwise tries to be self-

explanatory with sufficient descriptions of the intended behavior. Similarly, configuration data value constraints and default values, where used, are based on recommendations in current standards documentation. Since implementations vary widely in this respect, this version of the model specifies only a limited set of defaults and ranges with the expectation of being more prescriptive in future versions based on actual operator use.

Note that this version of the model is a work-in-progress in several respects. Although we present a complete framework for MPLS and traffic engineering from an operational perspective, some signaling protocol configuration will be completed in future revisions.

2. Model overview

The overall MPLS model is defined across several YANG modules and submodules but at a high level is organized into 4 main sections:

- o global -- configuration affecting MPLS behavior which exists independently of the underlying signaling protocol or label switched path configuration.

- o te-global-attributes -- configuration affecting MPLS-TE behavior which exists independently of the underlying signaling protocol or label switched path configuration.

- o signaling protocols -- configuration specific to signaling protocols used to setup and manage label switched paths.

- o label switched paths -- configuration specific to instantiating and managing individual label switched paths.

The top level of the model is shown in the tree view below:

```
+--rw mpls!
      +--rw global
      |     ...
      +--rw te-global-attributes
      |     ...
      +--rw signaling-protocols
      |     ...
      +--rw lsps
            ...
```

2.1. MPLS global

The global section of the framework provides configuration data for MPLS items which exist independently of an individual label switched path or signaling protocol and are applicable to the MPLS protocol itself. Items such as the depth of the label stack supported, or specific label ranges may be included here.

2.2. TE global attributes

The TE global attributes section of the framework provides configuration control for MPLS-TE items which exist independently of an individual label switched path or signaling protocol. These standalone items are applicable to the entire logical routing device, and establish fundamental configuration such as the threshold for interface bandwidth change that triggers update events into the IGP traffic engineering database (TED). Timers are also specified which determine the length of time an LSP must be present before being considered viable for forwarding use (mpls-lsp-install-delay), and the length of time between LSP teardown and removal of the LSP from the network element’s forwarding information base (mpls-lsp-cleanup-delay). Also specified are the name to value mappings of MPLS administrative groups (mpls-admin-groups).

```
+--rw te-global-attributes
      |  +--rw ted-update-threshold
      |     ...
      |  +--rw te-interfaces* [interface-name]
      |  |  +--rw interface-name                    string
      |  |  +--rw interface-admin-groups*           leafref
      |  |  +--rw interface-ted-update-threshold?   leafref
      |  +--rw te_lsp_timers
      |  |  +--rw config
      |  |  |  +--rw te-lsp-install-delay?      uint16
      |  |  |  +--rw te-lsp-cleanup-delay?      uint16
      |  |  |  +--rw te-lsp-reoptimize-timer?   uint16
      |  |  +--ro state
      |     ...
      |  +--rw mpls-admin-groups* [admin-group-name]
      |     +--rw admin-group-name     string
      |     +--rw admin-group-value?   uint32
```

2.3. Signaling protocol overview

The signaling protocol section of the framework provides configuration elements for configuring three major methods of signaling label switched paths: RSVP-TE, segment routing, and label

distribution protocol (LDP). BGP-LU will be included in a future version of this draft by definitions in the BGP model ([I-D.shaikh-idr-bgp-model]) and corresponding augmentations to the MPLS model.

```
+--rw signaling-protocols
      |  +--rw rsvp-te
      |     ...
      |  +--rw segment-routing
      |     ...
      |  +--rw ldp
      |     ...
```

Configuration of RSVP-TE is centered around interfaces on the device which participate in the protocol. A key focus is to expose common RSVP-TE configuration parameters which are used to enhance scale and reliability (refresh-reduction, refresh-reduction-reliable). Items which are applicable globally in the RSVP-TE protocol such as graceful restart, soft preemption and various statistics are grouped into a global section under the protocol.

```
  +--rw signaling-protocols
        |  +--rw rsvp-te
        |  |  +--rw global
        |  |  |  +--rw graceful-restart
        |     ...
        |  |  |  +--rw soft-preemption
        |     ...
        |  |  |  +--rw statistics
        |     ...
        |  |  |     +--ro state
        |  |  |        +--ro protocol
        |  |  |        |  +--ro hello-sent?   yang:counter32
        |  |  |        |  +--ro hello-rcvd?   yang:counter32
        |  |  |        |  +--ro path-sent?    yang:counter32
        |  |  |        |  +--ro path-rcvd?    yang:counter32
        |  |  |        +--ro error
        |  |  |           +--ro authentication-failure?   yang:counter32
        |  |  |           +--ro PathErr?                  yang:counter32
        |  |  |           +--ro ResvErr?                  yang:counter32
        |  |  |           +--ro path-timeout?             yang:counter32
        |  |  |           +--ro resv-timeout?             yang:counter32
        |  |  |           +--ro rate-limit?               yang:counter32
        |  |  +--rw interfaces* [interface-name]
        |  |     +--rw interface-name      string
        |  |     +--rw protocol_options
        |  |     |  +--rw config
        |  |     |  |  +--rw hello-interval?               uint16
        |  |     |  |  +--rw refresh-reduction?            boolean
        |  |     |  |  +--rw refresh-reduction-reliable?   boolean
        |  |     |  +--ro state
        |     ...
        |  |     +--rw authentication
        |     ...
        |  |     +--rw subscription
        |     ...
        |  |     +--rw link-protection
        |  |        +--rw config
        |  |        |  +--rw enable?                     boolean
        |  |        |  +--rw link-protection-only?       boolean
        |  |        |  +--rw bypass-optimize-interval?   uint16
        |     ...
```

Containers for specifying signaling via segment routing and LDP are also present. Specific subelements will be added for those protocols, as well as for BGP labeled unicast, in the next revision.

2.4. LSP overview

This part of the framework contains LSP information. At the high level, LSPs are split into three categories: traffic-engineering-capable (constrained-path), non-traffic-engineered determined by the IGP (unconstrained-path), and hop-by-hop configured (static).

```
+--rw mpls!
      +--rw lsps
         +--rw constrained-path
         |     ...
         +--rw unconstrained-path
         |     ...
         +--rw static-lsps
               ...
```

The first two categories, constrained-path and unconstrained-path are the ones for which multiple signaling protocols exist, and are organized in protocol-specific and protocol-independent sections. For example, traffic-engineered (constrained path) LSPs may be set up using RSVP-TE or segment routing, and unconstrained LSPs that follow the IGP path may be signaled with LDP or with segment routing. IGP-determined LSPs may also be signaled by RSVP but this usage is not considered in the current version of the model.

A portion of the data model for constrained path traffic-engineered LSPs signaled with RSVP is shown below. The first part of the model is signaling-protocol independent. Attributes such as the path computation method, the constraints for the the path, the bandwidth allocated to it, and even the frequency of reoptimization are signaling-protocol independent. Protocol specific data, such as the setup and hold priorities for RSVP are specified in the protocol specific configuration.

```
+--rw mpls!
      +--rw lsps
         +--rw constrained-path
         |  +--rw paths
         |  |  +--rw path* [path-name]
         |  |  |  +--rw path-name    leafref
         |  ...
         |  |  +--rw hops* [address]
         |  |     +--rw address    leafref
         |  ...
         |  +--rw label-switched-path* [signaled-name]
         |     +--rw signaled-name                leafref
         |     +--rw config
         |     |  +--rw signaled-name?             string
```

```
         |     |  +--rw lsp-description?           string
         |     |  +--rw destination?               inet:ip-address
         |     |  +--rw te-lsp-reoptimize-timer?   uint16
         |     +--ro state
         |  ...
         |     +--rw path-computation-method
         |     |  +--rw path-computation?   identityref
         |     |  +--rw explicit-path
         |     |  |  +--rw path-name?   leafref
         |     |  +--rw queried-path
         |     |  |  +--rw path-computation-server?   inet:ip-address
         |     |  +--rw locally-computed
         |     |     +--rw use-cspf?          boolean
         |     |     +--rw cspf-tiebreaker?   cspf-tie-breaking
         |     +--rw path-attributes
         |     |  +--rw config
         |     |  |  +--rw (lsp-bandwidth)?
         |     |  |  |  +--:(explicit)
         |     |  |  |  |  +--rw set-bandwidth?    uint32
         |     |  |  |  +--:(auto)
         |     |  |  |     +--rw auto-bandwidth
         |  ...
         |     |  |  +--rw metric?           te-metric-type
         |  ...
         |     +--rw lsp-placement-constraints
         |     |  +--rw admin-groups
         |     |  |  +--rw exclude-groups
         |     |  |     +--rw config
         |     |  |     |  +--rw exclude-groups* [admin-group-name]
         |     |  |     |     +--rw admin-group-name    leafref
         |  ...
         |     |  +--rw include-any-groups
         |     |  |  +--rw config
         |     |  |  |  +--rw include-any-groups* [admin-group-name]
         |     |  |  |     +--rw admin-group-name    leafref
         |  ...
         |     |  +--rw include-all-groups
         |     |     +--rw include-any-groups* [admin-group-name]
         |     |     |  +--rw admin-group-name    leafref
         |     |     +--rw config
         |     |     |  +--rw include-all-groups* [admin-group-name]
         |     |     |     +--rw admin-group-name    leafref
         |  ...
         |     +--rw protection
         |     |  +--rw config
         |     |  |  +--rw protection-style-requested?   mpls-protection-style
         |  ...
         |     +--rw path-setup
```

```
        |        +--rw rsvp!
        |        |  +--rw path-specification
        |        |  +--rw tunnel
        |        |     +--rw config
        |        |     |  +--rw tunnel-type?   mplst:tunnel-type
        |        |     +--ro state
        |        |     |  +--ro tunnel-type?   mplst:tunnel-type
        |        |     +--rw p2p-lsp
        |        |     |  +--rw config
        |        |     |  |  +--rw setup-priority?    uint8
        |        |     |  |  +--rw hold-priority?     uint8
        |        |     |  |  +--rw retry-timer?       uint16
        |        |     |  |  +--rw destination?       inet:ip-address
        |        |     |  |  +--rw tunnel-local-id?   union
        |        |     |  |  +--rw soft-preemption?   boolean
        |  ...
```

Similarly, the partial model for non-traffic-engineered, or IGP-based, LSPs is shown below:

```
+--rw mpls!
      +--rw lsps
         +--rw unconstrained-path
            +--rw path-setup-protocol
               +--rw ldp!
               |     ...
               +--rw segment-routing!
                     ...
```

3. Example use cases

3.1. Traffic engineered p2p LSP signaled with RSVP

A possible scenario may be the establishment of a mesh of traffic-engineered LSPs where RSVP signaling is desired, and the LSPs use a local constrained path calculation to determine their path. These LSPs would fall into the category of a constrained-path LSP. The LSP will specify the path setup method as RSVP inside the path-setup container, indicating the LSP desires RSVP signaling. The LSP would be configured as locally-computed under the path-computation-method container, specifying the use of CSPF (use-cspf). Additional attributes such as bandwidth (explicit or auto) are available in the path-attributes container. The relevant parts of the model are shown below:

```
+--rw mpls!
      +--rw lsps
         +--rw constrained-path
         |  +--rw label-switched-path* [signaled-name]
         |     +--rw signaled-name                leafref
         |     +--rw config
         |     |  +--rw signaled-name?             string
         |     |  +--rw lsp-description?           string
         |     |  +--rw destination?               inet:ip-address

         |  ...
         |     +--rw path-computation-method
         |  ...
         |     |  +--rw locally-computed
         |     |     +--rw use-cspf?          boolean
         |     |     +--rw cspf-tiebreaker?   cspf-tie-breaking
         |     +--rw path-attributes
         |     |  +--rw config
         |     |  |  +--rw (lsp-bandwidth)?
         |     |  |  |  +--:(explicit)
         |     |  |  |  |  +--rw set-bandwidth?    uint32
         |  ...
         |     |  |  +--rw metric?           te-metric-type
         |  ...
         |     +--rw protection
         |     |  +--rw config
         |     |  |  +--rw protection-style-requested?   mpls-protection-style
         |  ...
         |     +--rw path-setup
         |        +--rw rsvp!
         |        |  +--rw path-specification
         |        |  +--rw tunnel
         |        |     +--rw config
         |        |     |  +--rw tunnel-type?   mplst:tunnel-type
         |  ...
         |        |     +--rw p2p-lsp
         |        |     |  +--rw config
         |        |     |  |  +--rw setup-priority?    uint8
         |        |     |  |  +--rw hold-priority?     uint8
         |        |     |  |  +--rw retry-timer?       uint16
         |        |     |  |  +--rw destination?       inet:ip-address
         |        |     |  |  +--rw tunnel-local-id?   union
         |        |     |  |  +--rw soft-preemption?   boolean
```

3.2. Traffic engineered LSP signaled with SR

A possible scenario may be the establishment of disjoint paths in a network where there is no requirement for per-LSP state to be held on midpoint nodes within the network, or RSVP-TE is unsuitable (as described in [I-D.ietf-spring-segment-routing-mpls] and [I-D.shakir-rtgwg-sr-performance-engineered-lsps]). Such LSPs fall in the constrained-path category. Similar to any other traffic engineered LSPs, the path computation method must be specified. Path attributes, such as the as lsp- placement-constraints (expressed as administrative groups) or metric must be defined. Finally, the path must be specified in a signaling- protocol specific manner appropriate for SR. The same configuration elements from the tree above apply in this case, except that path setup is done by the head-end by building a label stack, rather than signaled.

3.3. IGP-congruent LDP-signaled LSP

A possible scenario may be the establishment of a full mesh of LSPs. When traffic engineering is not an objective, no constraints are placed on the end-to-end path, and the best- effort path can be setup using LDP signaling simply for label distribution. The LSPs follow IGP-computed paths, and fall in the unconstrained-path category in the model. Protocol-specific configuration pertaining to the signaling protocol used, such as the FEC definition and metrics assigned are in the path- setup-protocol portion of the model.

The relevant part of the model for this case is shown below:

```
  +--rw mpls!
      +--rw lsps
         +--rw unconstrained-path
            +--rw path-setup-protocol
               +--rw ldp!
                  +--rw tunnel
                     +--rw tunnel-type?   mplst:tunnel-type
                     +--rw ldp-type?      enumeration
                     +--rw p2p-lsp
                     |  +--rw fec-address*   inet:ip-prefix
                     +--rw p2mp-lsp
                     +--rw mp2mp-lsp
```

A common operational issue encountered when using LDP is traffic blackholing under the following scenario: when an IGP failure occurs, LDP is not aware of it as these are two protocols running independently, resulting in traffic blackholing at the IGP failure point even though LDP is up and running. LDP-IGP synchronization [RFC5443] can be used to cost out the IGP failing point/segment to

avoid the blackholing issue. The LDP-IGP synchronization function will be incorporated in a future version of this document.

Note that targeted LDP sessions are not discussed in this use case, and will be incorporated as a separate use case in a future version of this document.

4. Security Considerations

MPLS configuration has a significant impact on network operations, and as such any related protocol or model carries potential security risks.

YANG data models are generally designed to be used with the NETCONF protocol over an SSH transport. This provides an authenticated and secure channel over which to transfer BGP configuration and operational data. Note that use of alternate transport or data encoding (e.g., JSON over HTTPS) would require similar mechanisms for authenticating and securing access to configuration data.

Most of the data elements in the configuration model could be considered sensitive from a security standpoint. Unauthorized access or invalid data could cause major disruption.

5. IANA Considerations

This YANG data model and the component modules currently use a temporary ad-hoc namespace. If and when it is placed on redirected for the standards track, an appropriate namespace URI will be registered in the IETF XML Registry" [RFC3688]. The MPLS YANG modules will be registered in the "YANG Module Names" registry [RFC6020].

6. YANG modules

The modules and submodules comprising the MPLS configuration and operational model are currently organized as depcited below.

```
                                  +-------+
               +----------------->| MPLS  |<--------------+
               |                  +-------+               |
               |                      ^                   |
               |                      |                   |
          +----+-----+      +--------+-------+     +-----+-----+
          | TE LSPs  |      | IGP-based LSPs |     |static LSPs|
          |          |      |                |     |           |
          +----------+      +----------------+     +-----------+
              ^  ^                     ^  ^
              |  +----------------+   |  +--------+
              |                   |   |           |
              |   +------+      +-+---+-+      +--+--+
              +---+ RSVP |      |SEGMENT|      | LDP |
                  +------+      |ROUTING|      +-----+
                                +-------+
```

The base MPLS module includes submodules describing the three different types of support LSPs, i.e., traffic-engineered (constrained-path), IGP congruent (unconstrained-path), and static. The signaling protocol specific parts of the model are described in separate modules for RSVP, segment routing, and LDP. As mentioned earlier, support for BGP labeled unicast is also planned in a future revision.

A module defining various reusable MPLS types is included, and these modules also make use of the standard Internet types, such as IP addresses, as defined in RFC 6991 [RFC6991].

6.1. MPLS base modules

```
<CODE BEGINS> file mpls.yang
module mpls {

  yang-version "1";

  // namespace
  namespace "http://openconfig.net/yang/mpls";

  prefix "mpls";


  // import some basic types
  import mpls-types { prefix mplst; }
  import mpls-rsvp { prefix rsvp; }
  import mpls-sr { prefix sr; }
  import mpls-ldp { prefix ldp; }
```

```
  // include submodules
  include mpls-te;
  include mpls-igp;
  include mpls-static;


  // meta
  organization "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  description
    "This module provides data definitions for configuration of
    Multiprotocol Label Switching (MPLS) and associated protocols for
    signaling and traffic engineering.

    RFC 3031: Multiprotocol Label Switching Architecture

    The MPLS / TE data model consists of several modules and
    submodules as shown below.  The top-level MPLS module describes
    the overall framework.  Three types of LSPs are supported:

    i) traffic-engineered (or constrained-path)

    ii) IGP-congruent (LSPs that follow the IGP path)

    iii) static LSPs which are not signaled

    The structure of each of these LSP configurations is defined in
    corresponding submodules.  Companion modules define the relevant
    configuration and operational data specific to key signaling
    protocols used in operational practice.


                              +-------+
             +---------------->| MPLS  |<--------------+
             |                 +-------+               |
             |                     ^                   |
             |                     |                   |
        +----+-----+      +--------+-------+     +-----+-----+
        | TE LSPs  |      | IGP-based LSPs |     |static LSPs|
        |          |      |                |     |           |
        +----------+      +----------------+     +-----------+
            ^  ^                     ^  ^
            |  +----------------+   |  +--------+
            |                   |   |           |
```

```
          |   +------+      +-+---+-+      +--+--+
          +---+ RSVP |      |SEGMENT|      | LDP |
              +------+      |ROUTING|      +-----+
                            +-------+
    ";

  revision "2014-12-12" {
    description
      "Initial revision";
    reference "TBD";
  }

  // extension statements

  // feature statements

  // identity statements


  // grouping statements

  grouping path-setup-common {
    description "common definitions for all signaling protocols";

    // TODO: not clear we really need this
    leaf path-setup-type {
      type identityref {
        base mplst:path-setup-protocol;
      }
      description "path setup protocol to use with the LSP";
    }
  }

  grouping mpls-administrative-groups {
    description
      "global level definitions for MPLS link admin groups";

    list mpls-admin-groups {

      key admin-group-name;
      description "configuration of value to name mapping for mpls
      affinities/admin-groups";

      leaf admin-group-name {
        type string;
        description "name for mpls admin-group";
      }
```

```
      leaf admin-group-value {
        type uint32;
        description "value for mpls admin-group";
      }

    }
  }

 grouping mpls-ted-update-threshold_config {
    description "Configuration options for traffic
    engineering database update thresholds.";
    leaf-list ted-update-threshold {
      type mplst:percentage;
      max-elements 16;
      description "stepped percentages of interface bandwidth change
        which trigger update events into the IGP
        traffic engineering database (TED)";
    }
  }
  grouping mpls-ted-update-threshold {
    description "Top level group for traffic engineering
    database flooding options";
    container ted-update-threshold {
      description "Interface bandwidth change percentages
      that trigger update events into the IGP traffic
      engineering database (TED)";
      container config {
        description "Configuration parameters for TED
        update threshold ";
        uses mpls-ted-update-threshold_config;
      }
      container state {
        description "State parameters for TED update threshold ";
        config false;
        uses mpls-ted-update-threshold_config;
      }
    }
  }


  grouping te_lsp_delay_config {
    leaf te-lsp-install-delay {
      type uint16 {
        range 0..3600;
      }
      units seconds;
      description "delay the use of newly installed te lsp for a
      specified amount of time.";
```

```
    }

    leaf te-lsp-cleanup-delay {
      type uint16;
      units seconds;
      description "delay the removal of old te lsp for a specified
      amount of time";
    }
  }

  grouping te-interfaces {
    description "global level definitions for interfaces on which TE is run";

    // TODO: this should be made a reference to an interface in the
    // interfaces model
    // TODO - should probably have as key the interface name, also
    // need an easy way to specify all interfaces and to exclude
    // interfaces.
    list te-interfaces {

      key interface-name;
      description "interfaces for which MPLS is enabled";

      leaf interface-name {
        type string;
        description "reference to interface name";
        // TODO: add ref to interface model
      }

      leaf-list interface-admin-groups {
        type leafref {
          path "/mpls:mpls/mpls:te-global-attributes/mpls:mpls-admin-groups/"
          + "mpls:admin-group-name";
        }
        description
          "list of configured admin-groups on the interface";
      }

      leaf interface-ted-update-threshold {
        type leafref {
          path "/mpls/te-global-attributes/ted-update-threshold/config/ted-updat
e-threshold";
        }
        description
          "ted update threshold on the interface";
      }


    }
```

```
    container te_lsp_timers {
      description "definition for delays associated with setup and cleanup of TE
 LSPs";
      container config {
        uses te_lsp_delay_config;
        uses te_lsp_reoptimize_config;
      }
      container state {
        config false;
        uses te_lsp_delay_config;
        uses te_lsp_reoptimize_config;
      }
    }

  }

container mpls {
    presence "top-level container for MPLS config and operational
    state";

    description "Anchor point for mpls configuration and operational
    data";

    container global {
      // entropy label support, label ranges will be added here in the future.
     description "general mpls configuration applicable to any type of LSP and s
ignaling protocol";
    }

    container te-global-attributes {
      description "traffic-engineering global attributes";

      uses mpls-ted-update-threshold;
      uses te-interfaces;
      uses mpls-administrative-groups;

    }

    container signaling-protocols {
      description "top-level signaling protocol configuration";

      uses rsvp:rsvp-global;
      uses sr:sr-global;
      uses ldp:ldp-global;
    }

    container lsps {
      description "LSP definitions and configuration";

      container constrained-path {
```

```
        description "traffic-engineered LSPs supporting different
        path computation and signaling methods";

        uses mpls-te-global;
        uses path-definitions;

        list label-switched-path {
          key signaled-name;
          description "list of defined TE LSPs";

          leaf signaled-name {
            type leafref {
              path "../config/signaled-name";
            }

            description "LSP name, also carried in signaling messages
            when appropriate";
          }

          uses te-lsp-common;
          uses te-lsp-setup;
        }
      }

      container unconstrained-path {
        description "LSPs that use the IGP-determined path, i.e., non
        traffic-engineered, or non constrained-path";

        uses igp-lsp-common;
        uses igp-lsp-setup;

      }

      container static-lsps {
      description "statically configured LSPs, without dynamic
        signaling";

        uses static-lsp-main;
      }
    }
  }

  // augment statements

  // rpc statements

  // notification statements
}
```

<CODE ENDS>

```
<CODE BEGINS> file mpls-types.yang
module mpls-types {

  yang-version "1";

  // namespace
  namespace "http://openconfig.net/yang/mpls-types";

  prefix "mplst";



  // meta
  organization "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  description
    "General types for MPLS / TE data model";

  revision "2015-02-01" {
    description
      "Initial revision";
    reference "TBD";
  }

  // extension statements

  // feature statements

  // identity statements

  // using identities rather than enum types to simplify adding new
  // signaling protocols as they are introduced and supported
  identity path-setup-protocol {
    description "base identity for supported MPLS signaling
    protocols";
  }

  identity path-setup-rsvp {
    base path-setup-protocol;
    description "RSVP-TE signaling protocol";
  }
```

```
identity path-setup-sr {
  base path-setup-protocol;
  description "Segment routing";
}

identity path-setup-ldp {
  base path-setup-protocol;
  description "lDP - RFC 5036";
}

// typedef statements

typedef percentage {
  type uint8 {
    range "0..100";
  }
  description
    "Integer indicating a percentage value";
}


typedef mpls-label {
  type union {
    type uint32 {
      range 16..1048575;
    }
    type enumeration {
      enum IPV4_EXPLICIT_NULL {
        value 0;
        description "valid at the bottom of the label stack,
        indicates that stack must be popped and packet forwarded
        based on IPv4 header";
      }
      enum ROUTER_ALERT {
        value 1;
        description "allowed anywhere in the label stack except
        the bottom, local router delivers packet to the local CPU
        when this label is at the top of the stack";
      }
      enum IPV6_EXPLICIT_NULL {
        value 2;
        description "valid at the bottom of the label stack,
        indicates that stack must be popped and packet forwarded
        based on IPv6 header";
      }
      enum IMPLICIT_NULL {
        value 3;
        description "assigned by local LSR but not carried in
```

```
            packets";
          }
          enum ENTROPY_LABEL_INDICATOR {
            value 7;
            description "Entropy label indicator, to allow an LSR
            to distinguish between entropy label and applicaiton
            labels RFC 6790";
          }
        }
      }
      description "type for MPLS label value encoding";
      reference "RFC 3032 - MPLS Label Stack Encoding";
    }

    typedef tunnel-type {
      type enumeration {
        enum P2P {
          description "point-to-point label-switched-path";
        }
        enum P2MP {
          description "point-to-multipoint label-switched-path";
        }
        enum MP2MP {
          description "multipoint-to-multipoint label-switched-path";
        }
      }
      description "defines the tunnel type for the LSP";
      reference
        "RFC 6388 - Label Distribution Protocol Extensions for
        Point-to-Multipoint and Multipoint-to-Multipoint Label Switched
        Paths
        RFC 4875 - Extensions to  Resource Reservation Protocol
        - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE
        Label Switched Paths (LSPs)";
    }

    // grouping statements

    // data definition statements

    // augment statements

    // rpc statements

    // notification statements

  }
  <CODE ENDS>
```

6.2. MPLS LSP submodules

```
<CODE BEGINS> file mpls-te.yang
submodule mpls-te {

  yang-version "1";

  belongs-to "mpls" {
    prefix "mpls";
  }


  // import some basic types
  import ietf-inet-types { prefix inet; }
  import mpls-types { prefix mplst; }
  import mpls-rsvp { prefix rsvp; }
  import mpls-sr { prefix sr; }


  // meta
  organization "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  description
    "Configuration related to constrained-path LSPs and traffic
    engineering.  These definitions are not specific to a particular
    signaling protocol or mechanism (see related submodules for
        signaling protocol-specific configuration).";

  revision "2014-07-07" {
    description
      "Initial revision";
    reference "TBD";
  }

  // extension statements

  // feature statements

  // identity statements

  // using identities for path comp method, though enums may also
  // be appropriate if we decided these are the primary computation
  // mechanisms in future.
  identity path-computation-method {
```

```
    description "base identity for supported path computation
      mechanisms";
  }

  identity locally-computed {
    base path-computation-method;
    description "indicates a constrained-path LSP in which the
      path is computed by the local LER";
  }

  identity externally-queried {
    base path-computation-method;
    description "constrained-path LSP in which the path is
      obtained by querying an external source, such as a PCE server";
  }

  identity explicitly-defined {
    base path-computation-method;
    description "constrained-path LSP in which the path is
      explicitly specified as a collection of strict or/and loose
      hops";
  }

  // typedef statements

  typedef mpls-hop-type {
    type enumeration {
      enum LOOSE {
        description "loose hop in an explicit path";
      }
      enum STRICT {
        description "strict hop in an explicit path";
      }
    }
    description "enumerated type for specifying loose or strict
      paths";
  }

  typedef te-metric-type {
    type union {
      type enumeration {
        enum IGP {
          description "set the LSP metric to track the underlying
            IGP metric";
        }
      }
      type uint32;
    }
```

```
    description "union type for setting the LSP TE metric to a
      static value, or to track the IGP metric";
  }

  typedef cspf-tie-breaking {
    type enumeration {
      enum RANDOM {
        description "CSPF calculation selects a random path among
          multiple equal-cost paths to the destination";
      }
      enum LEAST_FILL {
        description "CSPF calculation selects the path with greatest
          available bandwidth";
      }
      enum MOST_FILL {
        description "CSPF calculation selects the path with the least
          available bandwidth";
      }
    }
    default RANDOM;
    description "type to indicate the CSPF selection policy when
      multiple equal cost paths are available";
  }

  typedef mpls-protection-style {
    type enumeration {
      enum UNPROTECTED {
        description "no protection is desired for the lsp";
      }
      enum LINK-PROTECTION-REQUESTED {
        description "link protection is desired for the lsp";
      }
      enum LINK-NODE-PROTECTION-REQUESTED {
        description "node and link protection is desired for the lsp";
      }
    }
    default UNPROTECTED;
    description
      "Specifies the protection type for the LSP";
  }

  // grouping statements

  grouping te_lsp_reoptimize_config {
    leaf te-lsp-reoptimize-timer {
      type uint16;
      units seconds;
      description "frequency of reoptimization of a te lsp";
```

```
    }
  }

  grouping te-lsp-common {
    description "common definitions for traffic-engineered LSPs";

    container config {
      uses te-lsp-common_config;
      uses te_lsp_reoptimize_config;
    }

    container state {
      config false;
      uses te-lsp-common_config;
      uses te_lsp_reoptimize_config;
    }

    container path-computation-method {
      description "select and configure the way the LSP path is
        computed";

      leaf path-computation {
        type identityref {
          base path-computation-method;
        }
        description "path computation method to use with the LSP";
      }

      uses te-lsp-comp-explicit;
      uses te-lsp-comp-queried;
      uses te-lsp-comp-local;
    }

    container path-attributes {
      description "general path attribute settings for TE-LSP
        tunnels";

      container config {
        description "configuration relating to LSP bandwidth
        and metrics";
        uses te-lsp-bandwidth_config;
        uses te-lsp-metric_config;
      }

      container state {
        description "operational state relating to LSP bandwidth
        and metrics";
        config false;
```

```
        uses te-lsp-bandwidth_config;
        uses te-lsp-metric_config;
      }

      // XXX - no, this is also there for LDP - also removed the
      // reference to "igp metric" as this is going to be confusing,
      // unless we mandate for the LSP to have the same metric as the
      // Igp, which is going to be hard with some vendors
      // implementations.

    }


    container lsp-placement-constraints {
      description
        "constraints on lsp routing such as admin-groups";

      container admin-groups {
        description
          "Include/Exclude constraints for link affinities";

        container exclude-groups {
          container config {
            description "configuration specifying admin
            groups which must be strictly excluded in the
            LSP path";
            uses te-lsp-exclude-admin-group_config;
          }
          container state {
            description "operational state reflecting
            admin groups which must be strictly excluded";
            config false;
            uses te-lsp-exclude-admin-group_config;
          }
        }
      }
      container include-any-groups {
        container config {
          description "configuration specifying admin
          groups, any of which must be included in the
          LSP path";
          uses te-lsp-include-any-admin-group_config;
        }
        container state {
          description "operational state relating to admin
          groups, any of which must be included in the
          LSP path";
          config false;
```

```
          uses te-lsp-include-any-admin-group_config;
        }
      }
      container include-all-groups {
          uses te-lsp-include-any-admin-group_config;
        container config {
          description "configuration specifying admin
          groups, all of which must be included in the
          LSP path";
          uses te-lsp-include-all-admin-group_config;
        }
        container state {
          description "operational state relating to admin
          groups, all of which must be included in the
          LSP path";
          config false;
          uses te-lsp-include-all-admin-group_config;
        }
      }
    }

    container protection {
      description "failure protection properties for the LSP";

      container config {
        description "configuration stating which MPLS protection
        options will be requested by the LSP";
        uses te-lsp-protection_config;
      }

      container state {
        description "operational state reflecting which
        MPLS protection options will be requested by the LSP";
        config false;
        uses te-lsp-protection_config;
      }

    }
  }

  grouping te-lsp-common_config {
    leaf signaled-name {
      type string;
      description "LSP name, also carried in signaling messages
        when appropriate";
    }

    leaf lsp-description {
```

```
      type string;
      description "optional text description for the LSP";
    }
    leaf destination {
      type inet:ip-address;
      description "destination egress node for the LSP";
    }
  }
  //}
  // TODO - note that this is only currently defined for
  // RSVP-like entities

  grouping te-lsp-bandwidth_config {
    choice lsp-bandwidth {
      default explicit;
      description "select how bandwidth for the LSP will be
        specified and managed";
      case explicit {
        leaf set-bandwidth {
          type uint32;
          description "set bandwidth explicitly, e.g., using
            offline calculation";
        }
      }
      case auto {
        uses te-lsp-auto-bandwidth_config;
      }
    }
  }

  grouping te-lsp-auto-bandwidth_config {
    container auto-bandwidth {
      description "configure auto-bandwidth operation in
        which devices automatically adjust bandwidth to meet
        requirements";

      leaf enabled {
        type boolean;
        default false;
        description "enables mpls auto-bandwidth on the
          lsp";
      }

      leaf min-bw {
        type uint32;
        description "set the minimum bandwidth in Mbps for an
          auto-bandwidth LSP";
      }
```

```
      leaf max-bw {
        type uint32;
        description "set the maximum bandwidth in Mbps for an
          auto-bandwidth LSP";
      }

      leaf adjust-interval {
        type uint32;
        description "time in seconds between adjustments to
          LSP bandwidth";
      }

      leaf adjust-threshold {
        type mplst:percentage;
        description "percentage difference between the LSP's
          specified bandwidth and its current bandwidth
          allocation -- if the difference is greater than the
          specified percentage, auto-bandwidth adjustment is
          triggered";
      }

      container overflow {
        description "configuration of MPLS overflow bandwidth
          adjustement for the LSP";
        uses te-lsp-overflow_config;
      }

      container underflow {
        description "configuration of MPLS underflow bandwidth
          adjustement for the LSP";
        uses te-lsp-underflow_config;
      }
    }
  }

  grouping te-lsp-metric_config {
    leaf metric {
      type te-metric-type;
      description "LSP metric, either explicit or IGP";
    }
  }

  grouping  te-lsp-overflow_config {
    description "configuration for mpls lsp bandwidth
      overflow adjustment";

    leaf enabled {
      type boolean;
```

```
      default false;
      description "enables mpls lsp bandwidth overflow
        adjustment on the lsp";
    }

    leaf overflow-threshold {
      type mplst:percentage;
      description "bandwidth percentage change to trigger
        an overflow event";

    }

    leaf trigger-event-count {
      type uint16;
      description "number of consecutive overflow sample
        events needed to trigger an overflow adjustment";
    }
  }

  grouping te-lsp-underflow_config {
    description
      "configuration for mpls lsp bandwidth
      underflow adjustment";

    leaf enabled {
      type boolean;
      default false;
      description "enables bandwidth underflow
        adjustment on the lsp";
    }

    leaf underflow-threshold {
      type mplst:percentage;
      description "bandwidth percentage change to trigger
        and underflow event";
    }

    leaf trigger-event-count {
      type uint16;
      description "number of consecutive underflow sample
        events needed to trigger an underflow adjustment";
    }
  }

  grouping te-lsp-exclude-admin-group_config {
    list exclude-groups {

      key admin-group-name;
```

```
      description
        "list of admin-groups to exclude in path calculation";

      leaf admin-group-name {
        type leafref {
          path "/mpls/te-global-attributes/mpls-admin-groups/" +
            "admin-group-name";
        }
        description
          "name of the admin group -- references a defined admin
          group";
      }
    }
  }

  grouping te-lsp-include-all-admin-group_config {
    list include-all-groups {

      key admin-group-name;
      description
        "list of admin-groups of which all must be included";

      leaf admin-group-name {
        type leafref {
          path "/mpls/te-global-attributes/mpls-admin-groups/" +
            "admin-group-name";
        }
        description
          "name of the admin group -- references a defined
          admin group";
      }
    }
  }

  grouping te-lsp-include-any-admin-group_config {
    list include-any-groups {

      key admin-group-name;
      description
        "list of admin-groups of which one must be included";

      leaf admin-group-name {
        type leafref {
          path "/mpls/te-global-attributes/mpls-admin-groups/" +
            "admin-group-name";
        }
        description
          "name of the admin group -- references a defined
```

```
          admin group";
      }
    }
  }

  grouping te-lsp-protection_config {
    leaf protection-style-requested {
      type mpls-protection-style {
      }
      description "style of mpls frr protection desired. both
        facility backup and one-to-one are options";
    }
  }

  grouping te-lsp-comp-explicit {
    description "definitions for LSPs in which hops are explicitly
      specified";

    container explicit-path {
      description "LSP with explicit path specification";

      leaf path-name {
        type leafref {
          path "/mpls/lsps/constrained-path/"
            + "paths/path/config/path-name";
          require-instance true;
        }
          description "reference to a defined path";
      }
    }
  }

  grouping te-lsp-comp-queried {
    description "definitons for LSPs computed by querying a remote
      service, e.g., PCE server";

    container queried-path {
      description "LSP with path queried from an external server";

      leaf path-computation-server {
      type inet:ip-address;
      description "Address of the external path computation
        server";
      }
    }
  }

  grouping te-lsp-comp-local {
```

```
    description "definitons for locally-computed LSPs";

    container locally-computed {
      description "LSP with path computed by local ingress LSR";

      leaf use-cspf {
        type boolean;
        description "Flag to enable CSPF for locally computed LSPs";
      }
      leaf cspf-tiebreaker {
        type cspf-tie-breaking;
        description
          "Determine the tie-breaking method to choose between
          equally desirable paths during CSFP computation";
      }
    }
  }


  grouping path-definitions-old {
    description "describe path configuration for specifying LSP
      hops";

    container paths {

      leaf path-name {
        type leafref {
          path "/mpls/lsps/constrained-path/"
            + "path-name/config/path/path-name";
          require-instance true;
        }
      }

      container named-path {
        description "definition for name of LSP path object";
        container config {
          uses te-lsp-path-name_config;
        }
        container state {
          config false;
          uses te-lsp-path-name_config;
        }
      }

      container hops {
        description "definition of hop objects for a LSP
          path";
        container config {
```

```
          uses te-lsp-path-hop_config;
        }
        container state {
          config false;
          uses te-lsp-path-hop_config;
        }
      }
    }
  }

  grouping path-definitions {
    description "common information for MPLS path definition";
    container paths {
      list path {
        description "definition for naming a LSP path
          object";
        key path-name;

        leaf path-name {
          type leafref {
            path "/mpls/lsps/constrained-path/"
            + "paths/path/config/path-name";
            require-instance true;
          }
        }
        container config {
          description "configuration for LSP path name";
          uses te-lsp-path-name_config;
        }
        container state {
          description "operational state for LSP path name";
          config false;
          uses te-lsp-path-name_config;
        }
      }
      list hops {
        description "definition of hop objects for a LSP
          path";
        key address;

        leaf address {
          type leafref {
            path "/mpls/lsps/constrained-path/"
              + "paths/hops/config/address";
            require-instance true;
          }
        }
        container config {
```

```
          description "configuration for specifying LSP path
            hops";
          uses te-lsp-path-hop_config;
        }
        container state {
          description "operational state for specifying LSP path
            hops";
          config false;
          uses te-lsp-path-hop_config;
        }
      }
    }
  }

  grouping te-lsp-path-information_config {
    description "common information for MPLS path definition";
    // TODO - this has to be redone with pathname in the config container and a
leafref
    list path {
      key path-name;
      description "specification of LSP path";

      leaf path-name {
        type string;
        description "identifier for the LSP path";
      }

      list hop {
        key address;
        description "specification of the strict and loose hops in
          the path";

        leaf address {
          type inet:ip-address;
          description "router hop for the LSP path";
        }

        leaf type {
          type mpls-hop-type;
          description "strict or loose hop";
        }
      }
    }
  }

  grouping te-lsp-path-name_config {

    description "common information for MPLS path name definition";
```

```
    leaf path-name {
      type string;
      description "identifier for the LSP path";
    }
  }

  grouping te-lsp-path-hop_config {

    description "common information for MPLS path hops definition";

    leaf address {
      type inet:ip-address;
      description "router hop for the LSP path";
    }

    leaf type {
      type mpls-hop-type;
      description "strict or loose hop";
    }
  }

  grouping te-lsp-setup {
    description "signaling protocol configuration for traffic
      engineered LSPs";

    container path-setup {
      description "select and configure the signaling method for
        the LSP";

      // uses path-setup-common;
      uses rsvp:te-lsp-rsvp-setup;
      uses sr:te-lsp-sr-setup;
    }
  }

  grouping mpls-te-global {
    description "global level defintions for mpls traffic
      engineered LSPs";
  }

// data definition statements

// augment statements

// rpc statements

// notification statements
}
```

<CODE ENDS>

```
<CODE BEGINS> file mpls-igp.yang
submodule mpls-igp {

  yang-version "1";

  belongs-to "mpls" {
    prefix "mpls";
  }


  // import some basic types
  import mpls-ldp { prefix ldp; }
  import mpls-sr { prefix sr; }



  // meta
  organization "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  description
    "Configuration generic configuration parameters for IGP-congruent
    LSPs";

  revision "2014-07-07" {
    description
      "Initial revision";
    reference "TBD";
  }

  // extension statements

  // feature statements

  // identity statements

  // typedef statements

  // grouping statements


  grouping igp-lsp-common {
```

```
      description "common definitions for IGP-congruent LSPs";

      // container path-attributes {
      //  description "general path attribute settings for IGP-based
      //  LSPs";

      //}

    }


    grouping igp-lsp-setup {
      description "signaling protocol definitions for IGP-based LSPs";

      container path-setup-protocol {
        description "select and configure the signaling method for
            the LSP";

        // uses path-setup-common;
        uses ldp:igp-lsp-ldp-setup;
        uses sr:igp-lsp-sr-setup;
      }
    }


    // data definition statements

    // augment statements

    // rpc statements

    // notification statements

  }
  <CODE ENDS>


  <CODE BEGINS> file mpls-static.yang
  submodule mpls-static {

    yang-version "1";

    belongs-to "mpls" {
      prefix "mpls";
    }

    // import some basic types
    import mpls-types {prefix mplst; }
```

```
import ietf-inet-types { prefix inet; }


// meta
organization "OpenConfig working group";

contact
  "OpenConfig working group
  netopenconfig@googlegroups.com";

description
  "Defines static LSP configuration";

revision "2015-02-01" {
  description
    "Initial revision";
  reference "TBD";
}

// extension statements

// feature statements

// identity statements

// typedef statements

// grouping statements

grouping static-lsp-common {
  description "common definitions for static LSPs";

  leaf next-hop {
    type inet:ip-address;
    description "next hop IP address for the LSP";
  }

  leaf incoming-label {
    type mplst:mpls-label;
    description "label value on the incoming packet";
  }

  leaf push-label {
    type mplst:mpls-label;
    description "label value to push at the current hop for the
    LSP";
  }
}
```

```
     grouping static-lsp-main {
       description "grouping for top level list of static LSPs";


       list label-switched-path {
         key name;
         description "list of defined static LSPs";

         leaf name {
           type string;
           description "name to identify the LSP";
         }

         // TODO: separation into ingress, transit, egress may help
         // to figure out what exactly is configured, but need to
         // consider whether implementations can support the
         // separation
         container ingress {
           description "Static LSPs for which the router is an
             ingress node";

           uses static-lsp-common;
         }

         container transit {
           description "static LSPs for which the router is a
             transit node";

           uses static-lsp-common;
         }

         container egress {
           description "static LSPs for which the router is a
             egress  node";

           uses static-lsp-common;
         }
       }
     }

     // data definition statements

     // augment statements

     // rpc statements

     // notification statements
```

```
   }
   <CODE ENDS>
```

6.3. MPLS signaling protocol modules

```
<CODE BEGINS> file mpls-rsvp.yang
module mpls-rsvp {

  yang-version "1";

  // namespace
  namespace "http://openconfig.net/yang/rsvp";

  prefix "rsvp";

  // import some basic types
  import ietf-inet-types { prefix inet; }
  import mpls-types { prefix mplst; }


  // meta
  organization "OpenConfig working group";

  contact
    "OpenConfig working group
    netopenconfig@googlegroups.com";

  description
    "Configuration for RSVP-TE signaling, including global protocol
    parameters and LSP-specific configuration for constrained-path
    LSPs";

  revision "2015-04-22" {
    description
      "Initial revision";
    reference "TBD";
  }

  // extension statements

  // feature statements

  // identity statements

  // typedef statements

  // grouping statements
```

```
  grouping mpls-rsvp-soft-preemption_config {
    description "Configuration for MPLS soft preemption";
    leaf enable {
      type boolean;
      default false;
      description "Enables soft preemption on a node.";
    }

    leaf soft-preemption-timeout {
      type uint16 {
        range 0..max;
      }
      // The RFC actually recommends 30 seconds as default.
      default 0;
      description "Timeout value for soft preemption to revert
        to hard preemption";
      reference "RFC5712 MPLS-TE soft preemption";
    }
  }

  grouping mpls-rsvp-soft-preemption {
    description "Top level group for MPLS soft preemption";
    container soft-preemption {
      description "Protocol options relating to RSVP
      soft preemption";
      container config {
        description "Configuration parameters relating to RSVP
        soft preemption support";
        uses mpls-rsvp-soft-preemption_config;
      }
      container state {
        description "State parameters relating to RSVP
        soft preemption support";
        config false;
        uses mpls-rsvp-soft-preemption_config;
      }
    }
  }


  grouping mpls-rsvp-protocol_options_config {
    description "RSVP protocol options configuration.";
    leaf hello-interval {
      type uint16 {
        range 1000..max;
      }
      default 9000;
      units milliseconds;
```

```
      description "set the interval in ms between RSVP hello
        messages";
    }
    leaf refresh-reduction {
      type boolean;
      default true;
      description "enables all RSVP refresh reduction message
        bundling, RSVP message ID, reliable message delivery
        and summary refresh";
      reference "RFC 2961 RSVP Refresh Overhead Reduction
        Extensions";
    }

    leaf refresh-reduction-reliable {
      type boolean;
      default true;
      description "enables RSVP refresh reduction reliable
        delivery and message_ID";
      reference "RFC 2961 RSVP Refresh Overhead Reduction
        Extensions";
    }
  }

  grouping mpls-rsvp-protocol_options {
    description "Top level group for RSVP protocol options";
    // TODO: confirm that the described semantics are supported
    // on various implementations. Finer grain configuration
    // will be vendor-specific
    container protocol_options {
      container config {
        description "Configuration for RSVP refresh reduction";
        uses mpls-rsvp-protocol_options_config;
      }
      container state {
        description "State for RSVP refresh reduction";
        config false;
        uses mpls-rsvp-protocol_options_config;
      }
    }
  }

  grouping mpls-rsvp-subscription_config {
    description "RSVP subscription configuration";
    leaf subscription {
      type mplst:percentage;
      description "percentage of the interface bandwidth that
        RSVP can reserve";
    }
```

```
  }
  grouping mpls-rsvp-subscription {
    description "Top level group for RSVP subscription options";
    container subscription {
      description "Bandwidth percentage reservable by RSVP
      on an interface";
      container config {
        uses mpls-rsvp-subscription_config;
      }
      container state {
        config false;
        uses mpls-rsvp-subscription_config;
      }
    }
  }

  grouping mpls-rsvp-graceful-restart_config {
    description
      "Configuration parameters relating to RSVP Graceful-Restart";

    leaf enable {
      type boolean;
      default false;
      description "Enables graceful restart on the node.";
    }

    leaf restart-time {
      type uint32;
      description
        "Graceful restart time (seconds).";
      reference
        "RFC 5495: Description of the Resource
        Reservation Protocol - Traffic-Engineered
        (RSVP-TE) Graceful Restart Procedures";
    }
    leaf recovery-time {
      type uint32;
      description
        "RSVP state recovery time";
    }
  }

  grouping mpls-rsvp-graceful-restart {
    description "Top level group for RSVP graceful-restart
    parameters";
    container graceful-restart {
      container config {
        description "Configuration parameters relating to
```

```
        graceful-restart";
        uses mpls-rsvp-graceful-restart_config;
      }
      container state {
        config false;
        description "State information associated with
        RSVP graceful-restart";
        uses mpls-rsvp-graceful-restart_config;
      }
    }
  }

  grouping mpls-rsvp-authentication_config {
    description "RSVP authentication parameters container.";
    leaf enable {
      type boolean;
      default false;
      description "Enables RSVP authentication on the node.";
    }
    leaf authentication-key {
      type string {
        // Juniper supports 1..16, Cisco has a much bigger range, up to 60.
        length "1..32";
      }
      description
        "authenticate RSVP signaling
        messages";
      reference
        "RFC 2747: RSVP Cryptographic Authentication";
    }
  }

  grouping mpls-rsvp-authentication {
    description "Top level group for RSVP authentication, as per RFC2747";
    container authentication {
      container config {
        description "Configuration parameters relating
        to authentication";
        uses mpls-rsvp-authentication_config;
      }
      container state {
        config false;
        description "State information associated
        with authentication";
        uses mpls-rsvp-authentication_config;
      }
    }
  }
```

```
  grouping mpls-rsvp-link-protection_config {
    description "RSVP facility (link/node) protection configuration";
    leaf enable {
      type boolean;
      default false;
      description "Enables facility protection on the interface.";
    }
    leaf link-protection-only {
      type boolean;
      default false;
      description "disables node protection on this interface,
                   and forces only link protection";
    }

    leaf bypass-optimize-interval {
      type uint16;
      units seconds;
      description "interval between periodic optimization
        of the bypass LSPs";
      // note: this is interface specific on juniper
      // on iox, this is global. need to resolve.
    }
    // to be completed, things like enabling link protection,
    // optimization times, etc.
  }

  grouping mpls-rsvp-link-protection {
    description "Top level group for RSVP protection";
    container link-protection {
      description "link-protection (NHOP) related configuration";
      container config {
        description "Configuration for link-protection";
        uses mpls-rsvp-link-protection_config;
      }
      container state {
        description "State for link-protection";
        config false;
        uses mpls-rsvp-link-protection_config;
      }

    }
  }

  grouping mpls-rsvp-error-statistics {
    description "RSVP-TE packet statistics";
    container error {
      description "RSVP-TE error statistics";
      leaf authentication-failure {
```

```
        type yang:counter32;
        description
          "Authentication failure count";
      }

      leaf PathErr {
        type yang:counter32;
        description
          "Path error to client count";
      }

      leaf ResvErr {
        type yang:counter32;
        description
          "Resv error to client count";
      }

      leaf path-timeout {
        type yang:counter32;
        description
          "Path timeout count";
      }

      leaf resv-timeout {
        type yang:counter32;
        description
          "Resv timeout count";
      }

      leaf rate-limit {
        type yang:counter32;
        description
          "Count of packets that were rate limited";
      }

      // TODO - complete the other error statistics
    }
  }

  grouping mpls-rsvp-protocol-statistics {
    description "RSVP protocol statistics";
    container protocol {
      description "RSVP-TE protocol statistics";
      leaf hello-sent {
        type yang:counter32;
        description
          "Hello sent count";
      }
```

```
      leaf hello-rcvd {
        type yang:counter32;
        description
          "Hello received count";
      }

      leaf path-sent {
        type yang:counter32;
        description
          "Path sent count";
      }

      leaf path-rcvd {
        type yang:counter32;
        description
          "Path received count";
      }

      // TODO - To be completed the other packet statistics
    }
  }

  grouping mpls-rsvp-statistics {
    description "RSVP-TE statistics";
    container statistics {
      description "Statistics related to RSVP-TE";
      container  config {
        description "Configuration for RSVP-TE statistics";
      }
      container state {
        config false;
        description "State for RSVP-TE statistics on an interface";
        uses mpls-rsvp-protocol-statistics;
        uses mpls-rsvp-error-statistics;
      }
    }
  }

  grouping rsvp-global {
    description "Global RSVP protocol configuration";
    container rsvp-te {
      description "RSVP-TE global signaling protocol configuration";

      container global {
        uses mpls-rsvp-graceful-restart;
        uses mpls-rsvp-soft-preemption;
        // TODO - reconcile global and per-interface protocol-related statistics
        uses mpls-rsvp-statistics;
```

```
      }
      // interfaces, bw percentages, hello timers, etc goes here";

      list interfaces {
        key interface-name;
        description "list of per-interface RSVP configurations";

        // TODO: update to interface ref -- move to separate
        // augmentation.
        leaf interface-name {
          type string;
          description "references a configured IP interface";
        }
        uses mpls-rsvp-protocol_options;
        uses mpls-rsvp-authentication;
        uses mpls-rsvp-subscription;
        // uses mpls-rsvp-ted-update-threshold; /* this moved to the te-global a
ttributes stanza */
        uses mpls-rsvp-link-protection;

      }
    }
  }

  grouping mpls-rsvp-te-tunnel {
    description "definitions for RSVP-signaled LSP tunnel types,
                 .e.g., applicable to point-to-point LSPs";

    container tunnel {
      description "contains configuration stanzas for different LSP
        tunnel types (P2P, P2MP, etc.)";

      container config {
        description "configuration of overall tunnel parameters";
        uses mpls-rsvp-tunnel-type_config;
      }

      container state {
        description "state information for mpls tunnels";
        config false;
        uses mpls-rsvp-tunnel-type_config;
      }
      uses mpls-rsvp-p2p-lsp-top;
      uses mpls-rsvp-p2mp-lsp-top;
    }
  }

  grouping mpls-rsvp-p2mp-lsp-top {
    description "Top level grouping for P2MP LSPs";
```

```
    container p2mp-lsp {
      when "tunnel-type = 'P2MP'" {
        description "container is active when LSP tunnel type is
          point to multipoint";
      }
      description "properties of point-to-multipoint tunnels";
      container config {
        description "configuration of p2mp lsps";
        uses mpls-rsvp-tunnel-rsvp-p2mp_config;
      }
      container state {
        description "state information of p2mp lsps";
        config false;
        uses mpls-rsvp-tunnel-rsvp-p2mp_config;
      }
    }
  }

  grouping mpls-rsvp-p2p-lsp-top {
    description "Top level grouping for P2P LSPs";
    container p2p-lsp {
      when "tunnel-type = 'P2P'" {
        description "container active when LSP tunnel type is
          point to point";
      }
      description "properties of point-to-point tunnels";
      container config {
        description "configuration for p2p lsps";
        uses mpls-rsvp-tunnel-rsvp-p2p_config;
      }
      container state {
        description "state information for p2p lsps";
        config false;
        uses mpls-rsvp-tunnel-rsvp-p2p_config;
      }
    }
  }

  grouping mpls-rsvp-tunnel-type_config {
    description "Configuration of MPLS tunnel type";
    leaf tunnel-type {
      type mplst:tunnel-type;
      description "specifies the type of LSP, e.g., P2P or P2MP";
    }
  }

  grouping mpls-rsvp-tunnel-rsvp-p2mp_config {
    description "properties of point-to-multipoint tunnels";
```

```
    leaf-list destination {
      type inet:ip-address;
      description "list of destinations / egress nodes for the
        multipoint LSP tunnel";
    }

  }

  grouping mpls-rsvp-tunnel-rsvp-p2p_config {
    description "properties of point-to-point tunnels";
    leaf setup-priority {
      type uint8 {
        range 0..7;
      }
      default 7;
      description "preemption priority during LSP setup, lower is
        higher priority; default 7 indicates that LSP will not
        preempt established LSPs during setup";
      reference "RFC 3209 - RSVP-TE: Extensions to RSVP for LSP Tunnels";
    }

    leaf hold-priority {
      type uint8 {
        range 0..7;
      }
      default 0;
      description "preemption priority once the LSP is established,
                   lower is higher priority; default 0 indicates that other LSPs
                     will not preempt the LSPs once established";
      reference "RFC 3209 - RSVP-TE: Extensions to RSVP for LSP Tunnels";
    }

    leaf retry-timer {
      type uint16 {
        range 1..600;
      }
      units seconds;
      description "sets the time between attempts to establish the
        LSP";
    }
    leaf destination {
      type inet:ip-address;
      description "destination egress node for the LSP";
    }

    leaf tunnel-local-id {
      type union {
        type uint32;
```

```
        type string;
      }
      description "locally signficant optional identifier for the
        LSP; may be a numerical or string value";
    }

    leaf soft-preemption {
      type boolean;
      description "enables RSVP soft-preemption on this LSP";
      default false;
    }
  }

  grouping te-lsp-rsvp-setup {
    description "data definitions for RSVP-signalled LSPs";
    container rsvp {
      presence "Presence of this container sets the LSP to use
        RSVP signaling";
      description "Configuration for RSVP-signalled TE LSPs";
      container path-specification {
        description "Definition of primary/backup paths for purpose
          of signaling the LSP";
      }
      uses mpls-rsvp-te-tunnel;
    }
  }

  // data definition statements

  // augment statements

  // rpc statements

  // notification statements

}
<CODE ENDS>


   <CODE BEGINS> file mpls-sr.yang
   module mpls-sr {

     yang-version "1";

     // namespace
     namespace "http://openconfig.net/yang/sr";

     prefix "sr";
```

```
     // import some basic types
     import ietf-inet-types { prefix inet; }
     import mpls-types { prefix mplst; }

     // meta
     organization "OpenConfig working group";

     contact
       "OpenConfig working group
       netopenconfig@googlegroups.com";

     description
       "Configuration for MPLS with segment routing-based LSPs,
       including global parameters, and LSP-specific configuration for
       both constrained-path and IGP-congruent LSPs";

     revision "2014-07-07" {
       description
         "Initial revision";
       reference "TBD";
     }

     // extension statements

     // feature statements

     // identity statements

     // typedef statements

     grouping srgb_config {

       // Matches the "global" configuration options in
       // draft-litkowski-spring-yang...
       // TODO: request for this to be a separate
       // grouping such that it can be included.

       leaf lower-bound {
         type uint32;
         description
           "Lower value in the block.";
       }
       leaf upper-bound {
         type uint32;
         description
           "Upper value in the block.";
       }
       description
```

```
          "List of global blocks to be advertised.";
      }

      grouping srgb_state {
        description
          "State parameters relating to the SRGB";

        leaf size {
          type uint32;
          description
            "Number of indexes in the SRGB block";
        }
        leaf free {
          type uint32;
          description
            "Number of SRGB indexes that have not yet been allocated";
        }
        leaf used {
          type uint32;
          description
            "Number of SRGB indexes that are currently allocated";
        }

        // TODO: where do we put LFIB entries?

      }

      grouping adjacency-sid_config {
        description
          "Configuration related to an Adjacency Segment Identifier
          (SID)";

        // tuned from draft-litkowski-spring-yang
        // TODO: need to send a patch

        leaf-list advertise {
          type enumeration {
            enum "PROTECTED" {
              description
                "Advertise an Adjacency-SID for this interface, which is
                eligible to be protected using a local protection
                mechanism on the local LSR. The local protection
                mechanism selected is dependent upon the configuration
                of RSVP-TE FRR or LFA elsewhere on the system";
            }
            enum UNPROTECTED {
              description
                "Advertise an Adajcency-SID for this interface, which is
```

```
                explicitly excluded from being protected by any local
                protection mechanism";
            }
          }
          description
            "Specifies the type of adjacency SID which should be
            advertised for the specified entity.";
        }

        leaf-list groups {
          type uint32;
          description
            "Specifies the groups to which this interface belongs.
            Setting a value in this list results in an additional AdjSID
            being advertised, with the S-bit set to 1. The AdjSID is
            assumed to be protected";
        }
      }

      grouping interface_config {
        description
          "Configuration parameters relating to a Segment Routing
          enabled interface";

        leaf interface {
          type string;
          // TODO: this should be changed to a leafref.
          description
            "Reference to the interface for which segment routing
            configuration is to be applied.";
        }
      }

      // grouping statements

      grouping sr-global {
        description "global segment routing signaling configuration";

        container segment-routing {
          description "SR global signaling config";

          list srgb {
            key "lower-bound upper-bound";
            uses srgb_config;
            container config {
              description
                "Configuration parameters relating to the Segment Routing
                Global Block (SRGB)";
```

```
            uses srgb_config;
          }
          container state {
            config false;
            description
              "State parameters relating to the Segment Routing Global
              Block (SRGB)";
            uses srgb_config;
            uses srgb_state;
          }
          description
            "List of Segment Routing Global Block (SRGB) entries. These
            label blocks are reserved to be allocated as domain-wide
            entries.";
        }

        list interfaces {
          key "interface";
          uses interface_config;
          container config {
            description
              "Interface configuration parameters for Segment Routing
              relating to the specified interface";
            uses interface_config;
          }
          container state {
            config false;
            description
              "State parameters for Segment Routing features relating
              to the specified interface";
            uses interface_config;
          }
          container adjacency-sid {
            description
              "Configuration for Adjacency SIDs that are related to
              the specified interface";
            container config {
              description
                "Configuration parameters for the Adjacency-SIDs
                that are related to this interface";
              uses adjacency-sid_config;
            }
            container state {
              config false;
              description
                "State parameters for the Adjacency-SIDs that are
                related to this interface";
              uses adjacency-sid_config;
```

```
            }
          }
          description
            "List of interfaces with associated segment routing
            configuration";
        }
      }
    }

    grouping te-tunnel-sr_config {
      description
        "Configuration parameters relating to all SR-TE LSPs";

      leaf tunnel-type {
        type mplst:tunnel-type;
        description "specifies the type of LSP, e.g., P2P or P2MP";
      }

      leaf sid-selection-mode {
        type enumeration {
          enum "ADJ-SID-ONLY" {
            description
              "The SR-TE tunnel should only use adjacency SIDs
              to build the SID stack to be pushed for the LSP";
          }
          enum "MIXED-MODE" {
            description
              "The SR-TE tunnel can use a mix of adjacency
              and prefix SIDs to build the SID stack to be pushed
              to the LSP";
          }
        }
        default "MIXED-MODE";
        description
          "The restrictions placed on the SIDs to be selected by the
          calculation method for the SR-TE LSP";
      }

      leaf sid-protection-required {
        type boolean;
        default "false";
        description
          "When this value is set to true, only SIDs that are
          protected are to be selected by the calculating method
          for the SR-TE LSP.";
      }
    }
```

```
     grouping te-tunnel-sr_state {
       description
         "State parameters relating to an SR-TE tunnel";
       // placeholder
       // TODO: selected SID stack
     }

     grouping te-sr_config {
       description
         "Configuration parameters relating to an SR-TE LSP";

       leaf destination {
         type inet:ip-address;
         description
           "The destination IP for the SR-TE LSP";
       }
     }

     grouping te-sr_state {
       description
         "State parameters relating to SR-TE for the LSP";
       // placeholder
     }

     grouping te-tunnel-sr {
       description "defintiions for SR-signaled LSP tunnel types,
       .e.g., applicable to point-to-point LSPs";

       container config {
         description
           "Configuration parameters relating to SR-TE";
         uses te-sr_config;
       }
       container state {
         config false;
         description
           "State parameters relating to SR-TE";
         uses te-sr_config;
         uses te-sr_state;
       }

       container tunnel {
         description "contains configuration stanzas for different LSP
         tunnel types (P2P, P2MP, etc.)";

         container config {
           description
             "Configuration parameters relating to SR-TE for the
```

```
            tunnel";
          uses te-tunnel-sr_config;
        }
        container state {
          config false;
          description
            "State information relating to SR-TE for the tunnel";
          uses te-tunnel-sr_config;
          uses te-tunnel-sr_state;
        }

        container p2p-lsp {
          when "tunnel-type = 'P2P'" {
            description "container active when LSP tunnel type is
            point to point";
          }
          description "Config for point-to-point tunnels";

          // fill out the configuration details per segment
          // routing requrements
        }
      }
    }


    grouping te-lsp-sr-setup {
      description "data definitions for SR signaling";

      container segment-routing {

        presence "Presence of this container sets the LSP to use
        SR signaling";

        description "Configuration for signaling SR-based TE LSPs";

        uses te-tunnel-sr;
      }
    }

    grouping sr_fec-address_config {
      description
        "Configuration parameters relating to a FEC that is to be
        advertised by Segment Routing";

      leaf fec-address {
        type inet:ip-prefix;
        description
          "FEC that is to be advertised as part of the Prefix-SID";
```

```
      }
    }

    grouping sr_fec-prefix-sid_config {
      description
        "Configuration parameters relating to the nature of the
        Prefix-SID that is to be advertised for a particular FEC";

      leaf type {
        type enumeration {
          enum "INDEX" {
            description
              "Set when the value of the prefix SID should be specified
              as an off-set from the SRGB's zero-value. When multiple
              SRGBs are specified, the zero-value is the minimum
              of their lower bounds";
          }
          enum "ABSOLUTE" {
            description
              "Set when the value of a prefix SID is specified as the
              absolute value within an SRGB. It is an error to specify
              an absolute value outside of a specified SRGB";
          }
        }
        default "INDEX";
        description
          "Specifies how the value of the Prefix-SID should be
          interpreted - whether as an offset to the SRGB, or as an
          absolute value";
      }

      leaf node-flag {
        type boolean;
        description
          "Specifies that the Prefix-SID is to be treated as a Node-SID
          by setting the N-flag in the advertised Prefix-SID TLV in the
          IGP";
      }

      leaf last-hop-behavior {
        type enumeration {
          enum "EXPLICIT-NULL" {
            description
              "Specifies that the explicit null label is to be used
              when the penultimate hop forwards a labelled packet to
              this Prefix-SID";
          }
          enum "UNCHANGED" {
```

```
            description
              "Specicies that the Prefix-SID's label value is to be
              left in place when the penultimate hop forwards to this
              Prefix-SID";
          }
          enum "PHP" {
            description
              "Specicies that the penultimate hop should pop the
              Prefix-SID label before forwarding to the eLER";
          }
        }
        description
          "Configuration relating to the LFIB actions for the
          Prefix-SID to be used by the penultimate-hop";
      }
    }


    grouping igp-tunnel-sr {
      description "defintiions for SR-signaled, IGP-based LSP tunnel
      types";

      container tunnel {
        description "contains configuration stanzas for different LSP
        tunnel types (P2P, P2MP, etc.)";

        leaf tunnel-type {
          type mplst:tunnel-type;
          description "specifies the type of LSP, e.g., P2P or P2MP";
        }

        container p2p-lsp {
          when "tunnel-type = 'P2P'" {
            description "container active when LSP tunnel type is
            point to point";
          }
          description "properties of point-to-point tunnels";

          list fec {
            key "fec-address";
            uses sr_fec-address_config;

            description
              "List of FECs that are to be originated as SR LSPs";

            container config {
              description
                "Configuration parameters relating to the FEC to be
```

```
                advertised by SR";
              uses sr_fec-address_config;
            }
            container state {
              config false;
              description
                "Operational state relating to a FEC advertised by SR";
              uses sr_fec-address_config;
            }
            container prefix-sid {
              description
                "Parameters relating to the Prefix-SID
                used for the originated FEC";

              container config {
                description
                  "Configuration parameters relating to the Prefix-SID
                  used for the originated FEC";
                uses sr_fec-prefix-sid_config;
              }
              container state {
                config false;
                description
                  "Operational state parameters relating to the
                  Prefix-SID used for the originated FEC";
                uses sr_fec-prefix-sid_config;
              }
            }
          }
        }
      }
    }

    grouping igp-lsp-sr-setup {
      description "grouping for SR-IGP path setup for IGP-congruent
      LSPs";

      container segment-routing {

        presence "Presence of this container sets the LSP to use
        SR signaling";

        description "segment routing signaling extensions for
        IGP-confgruent LSPs";

        uses igp-tunnel-sr;

      }
```

```
     }

     // data definition statements

     // augment statements

     // rpc statements

     // notification statements

   }
   <CODE ENDS>


   <CODE BEGINS> file mpls-ldp.yang
   module mpls-ldp {

     yang-version "1";

     // namespace
     namespace "http://openconfig.net/yang/ldp";

     prefix "ldp";

     // import some basic types
     import ietf-inet-types { prefix inet; }
     import mpls-types { prefix mplst; }


     // meta
     organization "OpenConfig working group";

     contact
       "OpenConfig working group
       netopenconfig@googlegroups.com";

     description
       "Configuration of Label Distribution Protocol global and LSP-
       specific parameters for IGP-congruent LSPs";

     revision "2014-07-07" {
       description
         "Initial revision";
       reference "TBD";
     }

     // extension statements
```

```
     // feature statements

     // identity statements

     // typedef statements

     // grouping statements

      grouping ldp-global {
       description "global LDP signaling configuration";

       container ldp {
         description "LDP global signaling configuration";

         container timers {
           description "LDP timers";
         }
       }
     }


     grouping igp-tunnel-ldp {
       description "common defintiions for LDP-signaled LSP tunnel
       types";

       container tunnel {
         description "contains configuration stanzas for different LSP
         tunnel types (P2P, P2MP, etc.)";

         leaf tunnel-type {
           type mplst:tunnel-type;
             description "specifies the type of LSP, e.g., P2P or P2MP";
         }

         leaf ldp-type {
           type enumeration {
             enum BASIC {
               description "basic hop-by-hop LSP";
             }
             enum TARGETED {
               description "tLDP LSP";
             }
           }
           description "specify basic or targeted LDP LSP";
         }

         container p2p-lsp {
           when "tunnel-type = 'P2P'" {
```

```
            description "container active when LSP tunnel type is
            point to point";
          }

          description "properties of point-to-point tunnels";

          leaf-list fec-address {
            type inet:ip-prefix;
            description "Address prefix for packets sharing the same
            forwarding equivalence class for the IGP-based LSP";
          }
        }

        container p2mp-lsp {
          when "tunnel-type = 'P2MP'" {
            description "container is active when LSP tunnel type is
            point to multipoint";
          }

          description "properties of point-to-multipoint tunnels";

          // TODO: specify group/source, etc.

        }

        container mp2mp-lsp {
          when "tunnel-type = 'MP2MP'" {
            description "container is active when LSP tunnel type is
            multipoint to multipoint";
          }

          description "properties of multipoint-to-multipoint tunnels";

          // TODO: specify group/source, etc.

        }
      }
    }

    grouping igp-lsp-ldp-setup {
      description "grouping for LDP setup attributes";

      container ldp {

        presence "Presence of this container sets the LSP to use
        LDP signaling";

        description "LDP signaling setup for IGP-congruent LSPs";
```

```
         // include tunnel (p2p, p2mp, ...)

         uses igp-tunnel-ldp;

       }
     }

     // data definition statements

     // augment statements

     // rpc statements

     // notification statements

   }
   <CODE ENDS>
```

7. Contributing Authors

The following people contributed significantly to this document and are listed below:

Ina Minei
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US
Email: inaminei@google.com

Anees Shaikh
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US
Email: aashaikh@google.com

Phil Bedard
Cox Communications
Atlanta, GA 30319
US
Email: phil.bedard@cox.com

8. Acknowledgements

The authors are grateful for valuable contributions to this document and the associated models from: Ebben Aires, Deepak Bansal, Nabil Bitar, Feihong Chen, Mazen Khaddam.

9. References

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J., and E. Crabbe, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-01 (work in progress), May 2015.

[I-D.openconfig-netmod-model-structure]
Shaikh, A., Shakir, R., D'Souza, K., and L. Fang, "Operational Structure and Organization of YANG Models", draft-openconfig-netmod-model-structure-00 (work in progress), March 2015.

[I-D.openconfig-netmod-opstate]
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-00 (work in progress), March 2015.

[I-D.shaikh-idr-bgp-model]
Shaikh, A., Shakir, R., Patel, K., Hares, S., D'Souza, K., Bansal, D., Clemm, A., Alex, A., Jethanandani, M., and X. Liu, "BGP Model for Service Provider Networks", draft-shaikh-idr-bgp-model-02 (work in progress), June 2015.

[I-D.shakir-rtgwg-sr-performance-engineered-lsps]
Shakir, R., Vernals, D., and A. Capello, "Performance Engineered LSPs using the Segment Routing Data-Plane", draft-shakir-rtgwg-sr-performance-engineered-lsps-00 (work in progress), July 2013.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, March 2009.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

Authors' Addresses

Joshua George
Google
1600 Amphitheatre Pkwy
Mountain View, CA 94043
US

Email: jgeorge@google.com

Luyuan Fang
Microsoft
15590 NE 31st St
Redmond, WA 98052
US

Email: lufang@microsoft.com

Eric Osborne
Level 3

Email: eric.osborne@level3.com

Rob Shakir
BT
pp. C3L, BT Centre
81, Newgate Street
London EC1A 7AJ
UK

Email: rob.shakir@bt.com
URI: http://www.bt.com/

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2016

V. Beeram
Juniper Networks
T. Saad
R. Gandhi
Cisco Systems Inc
X. Liu
Ericsson
H. Shah
Ciena
X. Chen
Huawei Technologies
R. Jones
Brocade
B. Wen
Comcast

July 06, 2015

A YANG Data Model for Resource Reservation Protocol (RSVP)
draft-saad-teas-yang-rsvp-02

Abstract

This document defines a YANG data model for the configuration and management of RSVP Protocol. The model defines generic RSVP protocol building blocks that can be augmented and used by other RSVP extension models such as RVSP extensions to Traffic-Engineering (RSVP-TE). The model covers the RSVP protocol configuration, operational state, remote procedural calls, and event notifications data.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g. ReST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage the RSVP protocol. This model covers generic protocol building blocks that can be augmented and used by other RSVP extension models- such as extensions for signaling RSVP-TE packet (or other technology specific) Label Switched Paths (LSP)s.

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].

1.2. Tree Diagram

A simplified graphical representation of the data model is presented in each section of the model. The following notations are used for the YANG model data tree representation.

```
<status> <flags> <name> <opts> <type>

 <status> is one of:
   +  for current
   x  for deprecated
   o  for obsolete

 <flags> is one of:
   rw  for read-write configuration data
   ro  for read-only non-configuration data
   -x  for execution rpcs
   -n  for notifications

 <name> is the name of the node

If the node is augmented into the tree from another module, its name
is printed as <prefix>:<name>

 <opts> is one of:
   ? for an optional leaf or node
   ! for a presence container
   * for a leaf-list or list
   Brackets [<keys>] for a list's keys
   Curly braces {<condition>} for optional feature that make node
conditional
   Colon : for marking case nodes
   Ellipses ("...") subtree contents not shown

   Parentheses enclose choice and case nodes, and case nodes are also
   marked with a colon (":").

 <type> is the name of the type for leafs and leaf-lists.
```

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]

Table 1: Prefixes and corresponding YANG modules

1.4. Open Issues and Next Steps

This document covers YANG models for data pertaining to the base RSVP, the generic RSVP-TE, and the packet RSVP-TE protocols. The current revision of this draft covers configuration and state data, but future revisions are expected to cover data for RPCs, and notifications.

1.4.1. Module Hierarchy

During discussions, some of the RSVP features were debated whether they should be present in the base RSVP model or in extension RSVP model (e.g. RSVP-TE model) especially that some features were defined in RSVP extension drafts for GMPLS or RSVP-TE states. For example, the RSVP Hello extensions defined in [RFC3209] with extensions to RSVP for TE states. However, RSVP Hellos extension can also apply to non RSVP-TE states, and some vendor implementations, allow it to be enabled independent of RSVP-TE features.

1.4.2. Model Data Organization

Throughout the model, the approach described in [I-D.openconfig-netmod-opstate] is adopted to represent data pertaining to configuration intended state, applied state and derived state data elements. Each container in the model hold 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.

The decision to use this approach was made to better align with the MPLS consolidated model in [I-D.openconfig-mpls-consolidated-model], and maximize reusability of groupings defined in this document and allow for possible convergence between the two models.

1.4.3. State Data

Pure state data (for example, protocol derived data) can be modeled using two options:

- o Contained inside the read-write container, under the "state" sub-container, as shown in Figure 2

- o Contained inside a separate read-only container

The first option allows for reusing the same containers that hold configuration read-write data under a "config" sub-container, and by

adding the state data under the read-only "state" sub-container of the container. For ephemeral or purely derived states (e.g. RSVP sessions), and since in this case the state would hang off a read-write parent container, it will be possible to delete the parent container and removing such state.

The second option entails defining a new read-only parent container in the model (e.g. neighbors-state) that holds the data.

This revision of the draft adopts the first option. Further discussions on this topic are expected to close on the best choice to adopt.

2. Design Considerations

2.1. Base Model

The base model discussed in this section covers base RSVP [RFC2205], and enhancements that pertain to the base protocol operation. RSVP-TE [RFC3209] and other traffic-engineering specific enhancements have been deliberately left out of this model to enable users to configure just the base RSVP protocol features in scenarios where traffic-engineering is not enabled/required. The generic and packet specific RSVP traffic-engineering model is an augmentation to the RSVP base model and is discussed in this revision of the document the packet RSVP-TE model is presented in Section 5.

Currently, the RSVP-TE module is presented as part of this draft, and is mostly packet centric. It is expected that the RSVP-TE YANG model will move to a separate document in the next revision.

```
     TE basic       +---------+        ^: import
     module         | ietf-te |        o: augment
                    +---------+
                       |   o
                       |   |
                       v   |
                    +--------------+
     RSVP-TE module | ietf-rsvp-te |o . . .
                    +--------------+         \
                       ^   |                  \
                       |   o              +--------------------+
                    +-----------+         | ietf-rsvp-otn-te  |
     RSVP module    | ietf-rsvp |         +--------------------+
                    +-----------+            RSVP-TE with OTN
                                             extensions
                                            (shown for illustration
                                             not in this document)
```

Figure 1: Relationship of RSVP and RSVP-TE modules with other protocol modules

2.2. Feature Set

The model in this revision of the document does not aim to be feature complete. The primary intent is to cover a set of standard generic features (listed below) that are commonly in use.

- o Authentication ([RFC2747])
- o Refresh Reduction ([RFC2961])
- o Hellos ([RFC3209])
- o Graceful Restart ([RFC3473], [RFC5063])

2.3. Configuration Inheritance

The defined data model supports configuration inheritance for neighbors, and interfaces. Data elements defined in the main container (e.g. the container that encompasses the list of interfaces, or neighbors) are assumed to apply equally to all elements of the list, unless overridden explicitly for a certain element (e.g. interface). Vendors are expected to augment the above container(s) to provide the list of inheritance command for their implementations.

2.4. Vendor Configuration Models

There two main popular types of routing protocol configuration that vendors may support:

- o protocol centric - all the protocol related configuration is contained within the protocol itself. Configuration belonging to multiple instances of the protocol running in different routing-instances (e.g. VRFs) are contained under the default routing instance [I-D.ietf-netmod-routing-cfg]:

- o VRF centric - all the protocol related configuration for a routing- instance is contained within this routing-instance.

On-going discussions within the IETF community have converged on adopting the VRF centric approach. The proposed model in this document adheres to this conclusion.

3. Model Organization

This document defines three YANG data models that cover the RSVP base, RSVP-TE generic, and RSVP-TE MPLS packet data that cover the configuration, state, RPCs, and notifications properties. The relationship between the different modules is depicted in Figure 1.

data pertaining to the configuration and operation of parameters applicable globally, per interface, neighbor or session.

4. RSVP Base YANG Model

This section describes the RSVP base YANG data model. It covers base RSVP protocol data defined by RSVP [RFC2205], and enhancements that pertain to the base protocol operation.

The container "rsvp" is the top level container in this data model. The presence of this container is expected to enable RSVP protocol functionality.

The approach described in [I-D.openconfig-netmod-opstate] allows for modeling the intended and respective applied and derived state. The TE state data in this model falls into one of the following categpries:

- o State corresponding to applied configuration

- o State corrsponding to derived state, counters, stats, etc.

Data for such state is contained under the respective "state" sub-container of the intended object (e.g. interface) as shown in Figure 2.

```
module: ietf-rsvp
   +--rw rsvp!
      +--rw globals
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--rw interfaces
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--rw neighbors
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--rw sessions
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
   rpcs:
      +--x global-rpc
      +--x interfaces-rpc
      +--x neighbors-rpc
      +--x sessions-rpc
   notifications:
      +--n global-notif
```

```
         +--n interfaces-notif
         +--n neighbors-notif
         +--n sessions-notif
```

Figure 2: RSVP highlevel model view

The following subsections provide overview of the parts of the model pertaining to configuration and state data.

4.1. Configuration and State Data

Global Data

This branch of the data model covers global configuration and states that control RSVP protocol behavior.

```
module: ietf-rsvp
   +--rw rsvp!
      +--rw globals
      |  +--rw config
      |  +--ro state
      |  |  +--ro statistics
      |  |     +--ro discontinuity-time?   yang:date-and-time
      |  |     +--ro packet
      |  |     |  +--ro sent?         yang:counter32
      |  |     |  +--ro rcvd?         yang:counter32
      |  |     |  +--ro tx-dropped?   yang:counter32
      |  |     |  +--ro rx-dropped?   yang:counter32
      |  |     |  +--ro tx-error?     yang:counter32
      |  |     |  +--ro rx-error?     yang:counter32
      |  |     +--ro protocol
      |  |     |  +--ro ack-sent?                   yang:counter32
      |  |     |  +--ro ack-rcvd?                   yang:counter32
      |  |     |  +--ro bundle-sent?                yang:counter32
      |  |     |  +--ro bundle-rcvd?                yang:counter32
      |  |     |  +--ro hello-sent?                 yang:counter32
      |  |     |  +--ro hello-rcvd?                 yang:counter32
      |  |     |  +--ro integrity-challenge-sent?   yang:counter32
      |  |     |  +--ro integrity-challenge-rcvd?   yang:counter32
      |  |     |  +--ro integrity-response-sent?    yang:counter32
      |  |     |  +--ro integrity-response-rcvd?    yang:counter32
      |  |     |  +--ro notify-sent?                yang:counter32
      |  |     |  +--ro notify-rcvd?                yang:counter32
      |  |     |  +--ro path-sent?                  yang:counter32
      |  |     |  +--ro path-rcvd?                  yang:counter32
      |  |     |  +--ro path-err-sent?              yang:counter32
      |  |     |  +--ro path-err-rcvd?              yang:counter32
```

```
         |  |     |  +--ro path-tear-sent?            yang:counter32
         |  |     |  +--ro path-tear-rcvd?            yang:counter32
         |  |     |  +--ro resv-sent?                 yang:counter32
         |  |     |  +--ro resv-rcvd?                 yang:counter32
         |  |     |  +--ro resv-confirm-sent?         yang:counter32
         |  |     |  +--ro resv-confirm-rcvd?         yang:counter32
         |  |     |  +--ro resv-err-sent?             yang:counter32
         |  |     |  +--ro resv-err-rcvd?             yang:counter32
         |  |     |  +--ro resv-tear-sent?            yang:counter32
         |  |     |  +--ro resv-tear-rcvd?            yang:counter32
         |  |     |  +--ro summary-refresh-sent?      yang:counter32
         |  |     |  +--ro summary-refresh-rcvd?      yang:counter32
         |  |     |  +--ro unknown-recv?              yang:counter32
         |  |     +--ro error
         |  |        +--ro authentication?   yang:counter64
         |  |        +--ro checksum?         yang:counter64
         |  |        +--ro packet-len?       yang:counter64
         |  +--rw graceful-restart! {graceful-restart}?
         |     +--rw config
         |     |  +--rw restart-time?    uint32
         |     |  +--rw recovery-time?   uint32
         |     +--ro state
         |        +--ro restart-time?    uint32
         |        +--ro recovery-time?   uint32
```

4.1.1. Interface Data

This branch of the data model covers configuration and state elements relevant to one or all RSVP interfaces. Any data configuration applied at the "interfaces" container level are equally applicable to all interfaces - unless overridden by explicit configuration under a specific interface.

```
module: ietf-rsvp
   +--rw rsvp!
      +--rw interfaces
         |  +--rw config
         |  +--ro state
         |  |  +--ro statistics
         |  |     +--ro discontinuity-time?   yang:date-and-time
         |  |     +--ro packet
         |  |     |  +--ro sent?         yang:counter32
         |  |     |  +--ro rcvd?         yang:counter32
         |  |     |  +--ro tx-dropped?   yang:counter32
         |  |     |  +--ro rx-dropped?   yang:counter32
         |  |     |  +--ro tx-error?     yang:counter32
         |  |     |  +--ro rx-error?     yang:counter32
         |  |     +--ro protocol
```

```
          |  |     |  +--ro ack-sent?                   yang:counter32
          |  |     |  +--ro ack-rcvd?                   yang:counter32
          |  |     |  +--ro bundle-sent?                yang:counter32
          |  |     |  +--ro bundle-rcvd?                yang:counter32
          |  |     |  +--ro hello-sent?                 yang:counter32
          |  |     |  +--ro hello-rcvd?                 yang:counter32
          |  |     |  +--ro integrity-challenge-sent?   yang:counter32
          |  |     |  +--ro integrity-challenge-rcvd?   yang:counter32
          |  |     |  +--ro integrity-response-sent?    yang:counter32
          |  |     |  +--ro integrity-response-rcvd?    yang:counter32
          |  |     |  +--ro notify-sent?                yang:counter32
          |  |     |  +--ro notify-rcvd?                yang:counter32
          |  |     |  +--ro path-sent?                  yang:counter32
          |  |     |  +--ro path-rcvd?                  yang:counter32
          |  |     |  +--ro path-err-sent?              yang:counter32
          |  |     |  +--ro path-err-rcvd?              yang:counter32
          |  |     |  +--ro path-tear-sent?             yang:counter32
          |  |     |  +--ro path-tear-rcvd?             yang:counter32
          |  |     |  +--ro resv-sent?                  yang:counter32
          |  |     |  +--ro resv-rcvd?                  yang:counter32
          |  |     |  +--ro resv-confirm-sent?          yang:counter32
          |  |     |  +--ro resv-confirm-rcvd?          yang:counter32
          |  |     |  +--ro resv-err-sent?              yang:counter32
          |  |     |  +--ro resv-err-rcvd?              yang:counter32
          |  |     |  +--ro resv-tear-sent?             yang:counter32
          |  |     |  +--ro resv-tear-rcvd?             yang:counter32
          |  |     |  +--ro summary-refresh-sent?       yang:counter32
          |  |     |  +--ro summary-refresh-rcvd?       yang:counter32
          |  |     |  +--ro unknown-recv?               yang:counter32
          |  |     +--ro error
          |  |        +--ro authentication?   yang:counter64
          |  |        +--ro checksum?         yang:counter64
          |  |        +--ro packet-len?       yang:counter64
          |  +--rw signaling-parameters
          |  |  +--rw config
          |  |  |  +--rw refresh-interval?        uint32
          |  |  |  +--rw refresh-misses?          uint32
          |  |  |  +--rw checksum?                uint32
          |  |  |  +--rw patherr-state-removal?   empty
          |  |  +--ro state
          |  |     +--ro refresh-interval?        uint32
          |  |     +--ro refresh-misses?          uint32
          |  |     +--ro checksum?                uint32
          |  |     +--ro patherr-state-removal?   empty
          |  +--rw refresh-reduction {refresh-reduction}?
          |  |  +--rw config
          |  |  |  +--rw bundle-message-max-size?    uint32
          |  |  |  +--rw disable?                    empty
```

```
          |  |  |  +--rw reliable-ack-hold-time?     uint32
          |  |  |  +--rw reliable-ack-max-size?      uint32
          |  |  |  +--rw reliable-retransmit-time?   uint32
          |  |  |  +--rw reliable-srefresh?          empty
          |  |  |  +--rw summary-max-size?           uint32
          |  |  +--ro state
          |  |     +--ro bundle-message-max-size?    uint32
          |  |     +--ro disable?                    empty
          |  |     +--ro reliable-ack-hold-time?     uint32
          |  |     +--ro reliable-ack-max-size?      uint32
          |  |     +--ro reliable-retransmit-time?   uint32
          |  |     +--ro reliable-srefresh?          empty
          |  |     +--ro summary-max-size?           uint32
          |  +--rw rsvp-hellos {hellos}?
          |  |  +--rw config
          |  |  |  +--rw interface-based?   empty
          |  |  |  +--rw hello-interval?    uint32
          |  |  |  +--rw hello-misses?      uint32
          |  |  +--ro state
          |  |     +--ro interface-based?   empty
          |  |     +--ro hello-interval?    uint32
          |  |     +--ro hello-misses?      uint32
          |  +--rw authentication {authentication}?
          |  |  +--rw config
          |  |  |  +--rw lifetime?      uint32
          |  |  |  +--rw window-size?   uint32
          |  |  |  +--rw challenge?     empty
          |  |  |  +--rw retransmits?   uint32
          |  |  |  +--rw (authentication-type)?
          |  |  |     +--:(string)
          |  |  |     |  +--rw password?      string
          |  |  |     |  +--rw algorithm?     identityref
          |  |  |     +--:(key-chain)
          |  |  |        +--rw key-chain?     string
          |  |  +--ro state
          |  |     +--ro lifetime?      uint32
          |  |     +--ro window-size?   uint32
          |  |     +--ro challenge?     empty
          |  |     +--ro retransmits?   uint32
          |  |     +--ro (authentication-type)?
          |  |        +--:(string)
          |  |        |  +--ro password?      string
          |  |        |  +--ro algorithm?     identityref
          |  |        +--:(key-chain)
          |  |           +--ro key-chain?     string
          |  +--rw interface* [interface]
          |     +--rw interface               if:interface-ref
          |     +--rw config
```

```
          |     +--ro state
          |     |  +--ro statistics
          |     |     +--ro discontinuity-time?   yang:date-and-time
          |     |     +--ro packet
          |     |     |  +--ro sent?         yang:counter32
          |     |     |  +--ro rcvd?         yang:counter32
          |     |     |  +--ro tx-dropped?   yang:counter32
          |     |     |  +--ro rx-dropped?   yang:counter32
          |     |     |  +--ro tx-error?     yang:counter32
          |     |     |  +--ro rx-error?     yang:counter32
          |     |     +--ro protocol
          |     |     |  +--ro ack-sent?                   yang:counter32
          |     |     |  +--ro ack-rcvd?                   yang:counter32
          |     |     |  +--ro bundle-sent?                yang:counter32
          |     |     |  +--ro bundle-rcvd?                yang:counter32
          |     |     |  +--ro hello-sent?                 yang:counter32
          |     |     |  +--ro hello-rcvd?                 yang:counter32
          |     |     |  +--ro integrity-challenge-sent?   yang:counter32
          |     |     |  +--ro integrity-challenge-rcvd?   yang:counter32
          |     |     |  +--ro integrity-response-sent?    yang:counter32
          |     |     |  +--ro integrity-response-rcvd?    yang:counter32
          |     |     |  +--ro notify-sent?                yang:counter32
          |     |     |  +--ro notify-rcvd?                yang:counter32
          |     |     |  +--ro path-sent?                  yang:counter32
          |     |     |  +--ro path-rcvd?                  yang:counter32
          |     |     |  +--ro path-err-sent?              yang:counter32
          |     |     |  +--ro path-err-rcvd?              yang:counter32
          |     |     |  +--ro path-tear-sent?             yang:counter32
          |     |     |  +--ro path-tear-rcvd?             yang:counter32
          |     |     |  +--ro resv-sent?                  yang:counter32
          |     |     |  +--ro resv-rcvd?                  yang:counter32
          |     |     |  +--ro resv-confirm-sent?          yang:counter32
          |     |     |  +--ro resv-confirm-rcvd?          yang:counter32
          |     |     |  +--ro resv-err-sent?              yang:counter32
          |     |     |  +--ro resv-err-rcvd?              yang:counter32
          |     |     |  +--ro resv-tear-sent?             yang:counter32
          |     |     |  +--ro resv-tear-rcvd?             yang:counter32
          |     |     |  +--ro summary-refresh-sent?       yang:counter32
          |     |     |  +--ro summary-refresh-rcvd?       yang:counter32
          |     |     |  +--ro unknown-recv?               yang:counter32
          |     |     +--ro error
          |     |        +--ro authentication?   yang:counter64
          |     |        +--ro checksum?         yang:counter64
          |     |        +--ro packet-len?       yang:counter64
          |     +--rw signaling-parameters
          |     |  +--rw config
          |     |  |  +--rw refresh-interval?        uint32
          |     |  |  +--rw refresh-misses?          uint32
```

```
          |     |  |  +--rw checksum?                uint32
          |     |  |  +--rw patherr-state-removal?   empty
          |     |  +--ro state
          |     |     +--ro refresh-interval?        uint32
          |     |     +--ro refresh-misses?          uint32
          |     |     +--ro checksum?                uint32
          |     |     +--ro patherr-state-removal?   empty
          |     +--rw refresh-reduction {refresh-reduction}?
          |     |  +--rw config
          |     |  |  +--rw bundle-message-max-size?    uint32
          |     |  |  +--rw disable?                    empty
          |     |  |  +--rw reliable-ack-hold-time?     uint32
          |     |  |  +--rw reliable-ack-max-size?      uint32
          |     |  |  +--rw reliable-retransmit-time?   uint32
          |     |  |  +--rw reliable-srefresh?          empty
          |     |  |  +--rw summary-max-size?           uint32
          |     |  +--ro state
          |     |     +--ro bundle-message-max-size?    uint32
          |     |     +--ro disable?                    empty
          |     |     +--ro reliable-ack-hold-time?     uint32
          |     |     +--ro reliable-ack-max-size?      uint32
          |     |     +--ro reliable-retransmit-time?   uint32
          |     |     +--ro reliable-srefresh?          empty
          |     |     +--ro summary-max-size?           uint32
          |     +--rw rsvp-hellos {hellos}?
          |     |  +--rw config
          |     |  |  +--rw interface-based?   empty
          |     |  |  +--rw hello-interval?    uint32
          |     |  |  +--rw hello-misses?      uint32
          |     |  +--ro state
          |     |     +--ro interface-based?   empty
          |     |     +--ro hello-interval?    uint32
          |     |     +--ro hello-misses?      uint32
          |     +--rw authentication {authentication}?
          |        +--rw config
          |        |  +--rw lifetime?      uint32
          |        |  +--rw window-size?   uint32
          |        |  +--rw challenge?     empty
          |        |  +--rw retransmits?   uint32
          |        |  +--rw (authentication-type)?
          |        |     +--:(string)
          |        |     |  +--rw password?      string
          |        |     |  +--rw algorithm?     identityref
          |        |     +--:(key-chain)
          |        |        +--rw key-chain?     string
          |        +--ro state
          |           +--ro lifetime?      uint32
          |           +--ro window-size?   uint32
```

```
         |           +--ro challenge?     empty
         |           +--ro retransmits?   uint32
         |           +--ro (authentication-type)?
         |              +--:(string)
         |              |  +--ro password?      string
         |              |  +--ro algorithm?     identityref
         |              +--:(key-chain)
         |                 +--ro key-chain?     string
```

4.1.2. Session Data

This branch of the data model covers configuration of elements relevant to RSVP neighbors. This would be discussed in detail in future revisions.

```
module: ietf-rsvp
   +--rw rsvp!
      +--rw sessions
         |  +--rw session* [src_port dst_port source destination]
         |     +--rw src_port       uint16
         |     +--rw dst_port       uint16
         |     +--rw source         inet:ip-address
         |     +--rw destination    inet:ip-address
         |     +--rw config
         |     +--ro state
```

4.1.3. Neighbor Data

This branch of the data model covers configuration of elements relevant to RSVP sessions. This would be discussed in detail in future revisions.

```
 module: ietf-rsvp
    +--rw rsvp!
       +--rw neighbors
          +--rw neighbor* [address]
             +--rw address                inet:ip-address
             +--rw neighbor-attributes
                +--rw config
                +--ro state
                   +--ro epoch?                      uint32
                   +--ro expiry-time?                uint32
                   +--ro graceful-restart
                   |  +--ro enabled?               boolean
                   |  +--ro local-restart-time?    uint32
                   |  +--ro local-recovery-time?   uint32
                   |  +--ro nbr-restart-time?      uint32
                   |  +--ro nbr-recovery-time?     uint32
                   |  +--ro helper-mode
                   |     +--ro helper-mode?                boolean
                   |     +--ro max-helper-restart-time?    uint32
                   |     +--ro max-helper-recovery-time?   uint32
                   |     +--ro nbr-restart-ttd?            uint32
                   |     +--ro nbr-recovery-ttd?           uint32
                   +--ro hello-status? enumeration {hellos}?
                   +--ro interface?                  if:interface-ref
                   +--ro neighbor-state?             enumeration
                   +--ro psb-count?                  uint32
                   +--ro rsb-count?                  uint32
                   +--ro refresh-reduction-capable?  boolean
                   +--ro restart-count?              uint32
                   +--ro restart-time?               yang:date-and-time
```

4.2. RPC and Notification Data

TBD.

4.3. YANG Module

```
<CODE BEGINS> file "ietf-rsvp@2015-07-06.yang"
module ietf-rsvp {

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp";

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

  /* import ietf-inet-types { prefix inet; } */
  import ietf-interfaces {
    prefix "if";
```

```
     }

     import ietf-inet-types {
       prefix inet;
     }

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

     organization
       "IETF TEAS Working Group";

     contact "TBA";

     description
       "This module contains the RSVP YANG data model.";

     revision 2015-07-06 {
       description "Latest revision of RSVP yang module.";
       reference "RFC2205";
     }

     identity hash-algorithm {
       description
         "Base identity for message-digest algorithm";
     }

     identity MD5 {
       base hash-algorithm;
       description
         "MD5 hash algorithm";
       reference "RFC1321";
     }

     identity SHA-1 {
       base hash-algorithm;
       description
         "SHA-1 hash algorithm";
       reference "NIST, FIPS PUB 180-1: Secure Hash Standard";
     }


     /* RSVP features */
     feature authentication {
       description
         "Indicates support for RSVP authentication";
     }
```

```
     feature error-statistics {
       description
         "Indicates support for error statistics";
     }

     feature global-statistics {
       description
         "Indicates support for global statistics";
     }

     feature graceful-restart {
       description
         "Indicates support for RSVP graceful restart";
     }

     feature hellos {
       description
         "Indicates support for RSVP hellos (RFC3209).";
     }

     feature notify {
       description
         "Indicates support for RSVP notify message (RFC3473).";
     }

     feature refresh-reduction {
       description
         "Indicates support for RSVP refresh reduction
         (RFC2961).";
     }

     feature per-interface-statistics {
       description
         "Indicates support for per interface statistics";
     }

     grouping graceful-restart_config {
       description
         "Configuration parameters relating to RSVP
         Graceful-Restart";
       leaf restart-time {
         type uint32;
         description
           "Graceful restart time (seconds).";
         reference
           "RFC 5495: Description of the Resource
           Reservation Protocol - Traffic-Engineered
           (RSVP-TE) Graceful Restart Procedures";
```

```
        }
        leaf recovery-time {
          type uint32;
          description
            "RSVP state recovery time";
        }
      }

      grouping graceful-restart {
        description
          "Top level grouping for RSVP graceful-restart
          parameters";
        container graceful-restart {
          if-feature graceful-restart;
          presence "Enable RSVP graceful restart on the node.";
          description
            "Top level container for RSVP graceful-restart";
          container config {
            description
              "Configuration parameters relating to
              graceful-restart";
            uses graceful-restart_config;
          }
          container state {
            config false;
            description
              "State information associated with RSVP
              graceful-restart";
            uses graceful-restart_config;
          }
        }
      }

      grouping authentication_config {
        description
          "Configuration parameters relating to RSVP
          authentication";
        leaf lifetime {
          type uint32 {
            range "30..86400";
          }
          description
            "Life time for each security association";
          reference
            "RFC 2747: RSVP Cryptographic
            Authentication";
        }
        leaf window-size {
```

```
          type uint32 {
            range "1..64";
          }
          description
            "Window-size to limit number of out-of-order
            messages.";
          reference
            "RFC 2747: RSVP Cryptographic
            Authentication";
        }
        leaf challenge {
          type empty;
          description
            "Enable challenge messages.";
          reference
            "RFC 2747: RSVP Cryptographic
            Authentication";
        }
        leaf retransmits {
          type uint32 {
            range "1..10000";
          }
          description
            "Number of retransmits when messages are
            dropped.";
          reference
            "RFC 2747: RSVP Cryptographic
            Authentication";
        }
        choice authentication-type {
          description
            "RSVP authentication choices";
          case string {
            leaf password {
              type string;
              description
                "An authentication key string";
            }
            leaf algorithm {
              type identityref {
                base hash-algorithm;
              }
              description
                "Cryptographic hash algorithm";
            }
          }
          case key-chain {
            description
```

```
            "Configure RSVP authentication.";
          leaf key-chain {
            type string {
              length "1..32";
            }
            description
              "Key chain name to authenticate RSVP
              signaling messages.";
            reference
              "RFC 2747: RSVP Cryptographic
              Authentication";
          }
        }
      }
    }

    grouping authentication {
      description
        "Top level grouping for RSVP authentication parameters";
      container authentication {
        if-feature authentication;
        description
          "Top level container for RSVP authentication
          parameters";
        container config {
          description
            "Configuration parameters relating to
            RSVP authentication";
          uses authentication_config;
        }
        container state {
          config false;
          description
            "State information associated with RSVP
            authentication";
          uses authentication_config;
        }
      }
    }

    grouping rsvp-hellos_config {
      description
        "Configuration parameters relating to RSVP
        hellos";
      leaf interface-based {
        type empty;
        description "Enable interface-based
          Hello adjacency if present.";
```

```
        }
        leaf hello-interval {
          type uint32 {
            range "3000..30000";
          }
          description
            "Configure interval between successive Hello
            messages in milliseconds.";
          reference
            "RFC 3209: RSVP-TE: Extensions to RSVP for
            LSP Tunnels.
            RFC 5495: Description of the Resource
            Reservation Protocol - Traffic-Engineered
            (RSVP-TE) Graceful Restart Procedures";
        }
        leaf hello-misses {
          type uint32 {
            range "1..10";
          }
          description
            "Configure max number of consecutive missed
            Hello messages.";
          reference
            "RFC 3209: RSVP-TE: Extensions to RSVP for
            LSP Tunnels RFC 5495: Description of the
            Resource Reservation Protocol - Traffic-
            Engineered (RSVP-TE) Graceful Restart
            Procedures";
        }
      }

      grouping rsvp-hellos {
        description
          "Top level grouping for RSVP hellos parameters";
        container rsvp-hellos {
          if-feature hellos;
          description
            "Top level container for RSVP hello parameters";
          container config {
            description
              "Configuration parameters relating to
              RSVP hellos";
            uses rsvp-hellos_config;
          }
          container state {
            config false;
            description
              "State information associated with RSVP
```

```
            hellos";
          uses rsvp-hellos_config;
        }
      }
    }

    grouping signaling-parameters_config {
      description
        "Configuration parameters relating to RSVP
        signaling";
      leaf refresh-interval {
        type uint32;
        description
          "Set interval between successive refreshes";
      }
      leaf refresh-misses {
        type uint32;
        description
          "Set max number of consecutive missed
          messages for state expiry";
      }
      leaf checksum {
        type uint32;
        description
          "Enable RSVP message checksum computation";
      }
      leaf patherr-state-removal {
        type empty;
        description
          "State-Removal flag in Path Error message
          if present.";
      }
    }

    grouping signaling-parameters {
      description
        "Top level grouping for RSVP signaling  parameters";
      container signaling-parameters {
        description
          "Top level container for RSVP signaling parameters";
        container config {
          description
            "Configuration parameters relating to
            RSVP signaling parameters";
          uses signaling-parameters_config;
        }
        container state {
          config false;
```

```
          description
            "State information associated with RSVP
            signaling parameters";
          uses signaling-parameters_config;
        }
      }
    }

    grouping interface-attributes {
      description
        "Top level grouping for RSVP interface properties";
      container config {
        description
          "Configuration parameters relating to
          RSVP interface parameters";
      }
      container state {
        config false;
        description
          "State information associated with RSVP
           interface parameters";
        uses statistics_state {
          if-feature per-interface-statistics;
        }
      }
    }

    grouping refresh-reduction_config {
      description
        "Configuration parameters relating to RSVP
        refresh reduction";
      leaf bundle-message-max-size {
        type uint32 {
          range "512..65000";
        }
        description
          "Configure maximum size (bytes) of a
          single RSVP Bundle message.";
      }
      leaf disable {
        type empty;
        description
          "Disable refresh reduction if present.";
      }
      leaf reliable-ack-hold-time {
        type uint32 {
          range "100..5000";
        }
```

```
          description
            "Configure hold time in milliseconds for
            sending RSVP ACK message(s).";
        }
        leaf reliable-ack-max-size {
          type uint32 {
            range "20..65000";
          }
          description
            "Configure max size of a single RSVP ACK
            message.";
        }
        leaf reliable-retransmit-time {
          type uint32 {
            range "100..10000";
          }
          description
            "Configure min delay in milliseconds to
            wait for an ACK before a retransmit.";
        }
        leaf reliable-srefresh {
          type empty;
          description
            "Configure use of reliable messaging for
            summary refresh if present.";
        }
        leaf summary-max-size {
          type uint32 {
            range "20..65000";
          }
          description
            "Configure max size (bytes) of a single
            RSVP summary refresh message.";
        }
      }

      grouping refresh-reduction {
        description
          "Top level grouping for RSVP refresh reduction
          parameters";
        container refresh-reduction {
          if-feature refresh-reduction;
          description
            "Top level container for RSVP refresh reduction
            parameters";
          container config {
            description
              "Configuration parameters relating to
```

```
            RSVP refresh reduction";
          uses refresh-reduction_config;
        }
        container state {
          config false;
          description
            "State information associated with RSVP
            refresh reduction";
          uses refresh-reduction_config;
        }
      }
    }

    grouping neighbor-derived_state {
      description
        "Derived state at neighbor level.";

      leaf epoch {
        type uint32;
        description
          "Neighbor epoch.";
      }

      leaf expiry-time {
        type uint32;
        units seconds;
        description
          "Neighbor expiry time after which the neighbor state
          is purged if no states associated with it";
      }

      container graceful-restart {
        description
          "Graceful restart information.";

        leaf enabled {
          type boolean;
          description
            "'true' if graceful restart is enabled for the
            neighbor.";
        }

        leaf local-restart-time {
          type uint32;
          units seconds;
          description
            "Local node restart time";
        }
```

```
         leaf local-recovery-time {
           type uint32;
           units seconds;
           description
             "Local node recover time";
         }

         leaf nbr-restart-time {
           type uint32;
           units seconds;
           description
             "Neighbor restart time";
         }

         leaf nbr-recovery-time {
           type uint32;
           units seconds;
           description
             "Neighbor recover time";
         }

         container helper-mode {
           description
             "Helper mode information ";

           leaf helper-mode {
             type boolean;
             description
               "'true' if helper mode is enabled.";
           }

           leaf max-helper-restart-time {
             type uint32;
             units seconds;
             description
               "The time the router or switch waits after it
                discovers that a neighboring router has gone down
                before it declares the neighbor down";
           }

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

```
          leaf nbr-restart-ttd {
            type uint32;
            units seconds;
            description
              "Number of seconds remaining for neighbor to send
               Hello message after restart.";
          }

          leaf nbr-recovery-ttd {
            type uint32;
            units seconds;
            description
              "Number of seconds remaining for neighbor to
              refresh.";
          }
        } // helper-mode
      } // graceful-restart

      leaf hello-status {
        if-feature hellos;
        type enumeration {
          enum "enabled" {
            description
              "Enabled";
          }
          enum "disabled" {
            description
              "Disabled";
          }
          enum "restarting" {
            description
              "Restarting";
          }
        }
        description
          "Hello status";
      }

      leaf interface {
        type if:interface-ref;
        description
          "Interface to this neighbor.";
      }

      leaf neighbor-state {
        type enumeration {
          enum "up" {
            description
```

```
              "up";
          }
          enum "down" {
            description
              "down";
          }
          enum "hello-disable" {
            description
              "hello-disable";
          }
          enum "restarting" {
            description
              "restarting";
          }
        }
        description
          "Neighbor state";
      }

      leaf psb-count {
        type uint32;
        description
          "Number of PSB state currently referencing the
          neighbor.";
      }

      leaf rsb-count {
        type uint32;
        description
          "Number of RSB state currently referencing the
          neighbor.";
      }

      leaf refresh-reduction-capable {
        if-feature refresh-reduction;
        type boolean;
        description
          "Whether neighbor is refresh reduction capable.";
      }

      leaf restart-count {
        type uint32;
        description
          "Number of times this neighbor restart";
      }

      leaf restart-time {
        type yang:date-and-time;
```

```
        description
          "Last restart time of the neighbor";
      }
    } // neighbor-derived_state

    grouping statistics_state {
      description "RSVP statistic attributes.";
      container statistics {
        description
          "RSVP 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.";
        }

        container packet {
          description
            "Packet statistics.";
          leaf sent {
            type yang:counter32;
            description
              "Packet sent count";
          }

          leaf rcvd {
            type yang:counter32;
            description
              "Packet received count";
          }

          leaf tx-dropped {
            type yang:counter32;
            description
              "Packet tx dropped count";
          }

          leaf rx-dropped {
            type yang:counter32;
            description
              "Packet rx dropped count";
          }
```

```
          leaf tx-error {
            type yang:counter32;
            description
              "Packet tx error count";
          }

          leaf rx-error {
            type yang:counter32;
            description
              "Packet rx error count";
          }
        }

        container protocol {
          description
            "RSVP protocol statistics.";
          leaf ack-sent {
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Hello sent count";
          }

          leaf ack-rcvd {
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Hello received count";
          }

          leaf bundle-sent {
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Bundle sent count";
          }

          leaf bundle-rcvd {
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Bundle received count";
          }

          leaf hello-sent {
            if-feature hellos;
            type yang:counter32;
            description
```

```
                "Hello sent count";
            }

            leaf hello-rcvd {
              if-feature hellos;
              type yang:counter32;
              description
                "Hello received count";
            }

            leaf integrity-challenge-sent {
              if-feature authentication;
              type yang:counter32;
              description
                "Integrity Challenge sent count";
            }

            leaf integrity-challenge-rcvd {
              if-feature authentication;
              type yang:counter32;
              description
                "Integrity Challenge received count";
            }

            leaf integrity-response-sent {
              if-feature authentication;
              type yang:counter32;
              description
                "Integrity Response sent count";
            }

            leaf integrity-response-rcvd {
              if-feature authentication;
              type yang:counter32;
              description
                "Integrity Response received count";
            }

            leaf notify-sent {
              if-feature refresh-reduction;
              type yang:counter32;
              description
                "Notify sent count";
            }

            leaf notify-rcvd {
              if-feature refresh-reduction;
              type yang:counter32;
```

```
            description
              "Notify received count";
          }

          leaf path-sent {
            type yang:counter32;
            description
              "Path sent count";
          }

          leaf path-rcvd {
            type yang:counter32;
            description
              "Path received count";
          }

          leaf path-err-sent {
            type yang:counter32;
            description
              "Path error sent count";
          }

          leaf path-err-rcvd {
            type yang:counter32;
            description
              "Path error received count";
          }

          leaf path-tear-sent {
            type yang:counter32;
            description
              "Path tear sent count";
          }

          leaf path-tear-rcvd {
            type yang:counter32;
            description
              "Path tear received count";
          }

          leaf resv-sent {
            type yang:counter32;
            description
              "Resv sent count";
          }

          leaf resv-rcvd {
            type yang:counter32;
```

```
            description
              "Resv received count";
          }

          leaf resv-confirm-sent {
            type yang:counter32;
            description
              "Confirm sent count";
          }

          leaf resv-confirm-rcvd {
            type yang:counter32;
            description
              "Confirm received count";
          }

          leaf resv-err-sent {
            type yang:counter32;
            description
              "Resv error sent count";
          }

          leaf resv-err-rcvd {
            type yang:counter32;
            description
              "Resv error received count";
          }

          leaf resv-tear-sent {
            type yang:counter32;
            description
              "Resv tear sent count";
          }

          leaf resv-tear-rcvd {
            type yang:counter32;
            description
              "Resv tear received count";
          }

          leaf summary-refresh-sent {
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Summary refresh sent count";
          }

          leaf summary-refresh-rcvd {
```

```
            if-feature refresh-reduction;
            type yang:counter32;
            description
              "Summary refresh received count";
          }

          leaf unknown-recv {
            type yang:counter32;
            description
              "Unknown packet received count";
          }
        } // rsvp

        container error {
          description
            "Error statistics.";

          leaf authentication {
            type yang:counter64;
            description
              "The total number of packets received with an
              authentication failure.";
          }

          leaf checksum {
            type yang:counter64;
            description
              "The total number of packets received with an invalid
              checksum value.";
          }

          leaf packet-len {
            type yang:counter64;
            description
              "The total number of packets received with an invalid
              packet length.";
          }
        } // error
      } // statistics
    } // statistics_state

    grouping global-attributes {
      description
        "Top level grouping for RSVP global properties";
      container config {
        description
          "Configuration globals properties";
      }
```

```
      container state {
        config false;
        description
          "State information associated with RSVP
           global properties";
        uses statistics_state {
          if-feature global-statistics;
        }
      }
    }

    grouping session-attributes {
      description
        "Top level grouping for RSVP session properties";
      container config {
        description
          "Configuration for session properties";
      }
      container state {
        config false;
        description
          "State information associated with RSVP
           session properties";
      }
    }

    grouping neighbor-attributes {
      description
        "Top level grouping for RSVP neighbor properties";
      container neighbor-attributes {
        description
          "Top level container for RSVP neighbor properties";
        container config {
          description
            "Configuration for neighbor properties";
        }
        container state {
          config false;
          description
            "State information associated with RSVP
             neighbor properties";
          uses neighbor-derived_state;
        }
      }
    }

    container rsvp {
      presence "Enable RSVP feature";
```

```
       description "RSVP feature container";
       container globals {
         description "RSVP global properties.";
         uses global-attributes;
         uses graceful-restart;
       }

       container interfaces {
         description
           "RSVP interfaces container";
         uses interface-attributes;
         uses signaling-parameters;
         uses refresh-reduction;
         uses rsvp-hellos;
         uses authentication;

         list interface {
           key "interface";
           description
             "RSVP interfaces.";
           leaf interface {
             type if:interface-ref;
             description
               "RSVP interface.";
           }
           uses interface-attributes;
           uses signaling-parameters;
           uses refresh-reduction;
           uses rsvp-hellos;
           uses authentication;
         }
       }

       container sessions {
         description
           "RSVP sessions container";
         list session {
           key "src_port dst_port source destination";
           description
             "List of RSVP sessions";
           leaf src_port {
             type uint16;
             description "RSVP source port";
             reference "RFC2205";
           }
           leaf dst_port {
             type uint16;
             description "RSVP destination port";
```

```
            reference "RFC2205";
          }
          leaf source {
            type inet:ip-address;
            description "RSVP source address";
            reference "RFC2205";
          }
          leaf destination {
            type inet:ip-address;
            description "RSVP destination address";
            reference "RFC2205";
          }
          uses session-attributes;
        }
      }

      container neighbors {
        description
          "RSVP neighbors container";
        list neighbor {
          key "address";
          description
            "List of RSVP neighbors";
          leaf address {
            type inet:ip-address;
            description
              "Neighbor address";
          }
          uses neighbor-attributes;
        }
      }
    }
  }
  <CODE ENDS>
```

5. RSVP-TE Generic YANG Model

This section contains the augmentation of the RSVP base YANG model for signalling Traffic-Engineering (RSVP-TE) Label Switched Paths (LSPs). New module is introduced that augment the RSVP-TE generic module to cover data items that are technology agnostic.

This model imports and augments the base RSVP YANG model (presented in Section 4.3). It also imports and augments the TE YANG model defined in [I-D.saad-teas-yang-te] to enable configuration of RSVP-TE attributes on TE tunnels.

The following subsections provide overview of the parts of the RSVP-TE generic model pertaining to configuration and state data.

5.1. Configuration and State Data

There are three types of configuration and state data nodes in this module:

- those augmenting or extending the base RSVP module
- those augmenting or extending the base TE generic module
- those that are specific to the RSVP-TE module

Below is a YANG tree representation for data items defined in the RSVP-TE generic module:

```
module: ietf-rsvp-te
augment /rsvp:rsvp/rsvp:globals:
augment /rsvp:rsvp/rsvp:interfaces:
augment /rsvp:rsvp/rsvp:interfaces/rsvp:interface:
augment /rsvp:rsvp/rsvp:sessions:
augment /rsvp:rsvp/rsvp:neighbors:
augment /ietf-te:te/ietf-te:tunnels/ietf-te:tunnel:
   +--rw config
   |  +--rw lsp-source?                  inet:ip-address
   |  +--rw lsp-signaled-name?           string
   |  +--rw lsp-priority-setup?          uint8
   |  +--rw lsp-priority-hold?           uint8
   |  +--rw local-recording-desired?     empty
   |  +--rw se-style-desired?            empty
   |  +--rw path-reevaluation-request?   empty
   |  +--rw soft-preemption-desired?     empty
   |  +--rw end-to-end-routing?          empty
   |  +--rw boundary-rerouting?          empty
   |  +--rw segment-based-rerouting?     empty
   |  +--rw lsp-integrety-required?      empty
   |  +--rw contiguous-lsp?              empty
   |  +--rw lsp-stitching-desired?       empty
   |  +--rw preplanned-lsp?              empty
   |  +--rw oob-mapping?                 empty
   +--ro state
      +--ro lsp-source?                  inet:ip-address
      +--ro lsp-signaled-name?           string
      +--ro lsp-priority-setup?          uint8
      +--ro lsp-priority-hold?           uint8
      +--ro local-recording-desired?     empty
      +--ro se-style-desired?            empty
```

```
         +--ro path-reevaluation-request?   empty
         +--ro soft-preemption-desired?     empty
         +--ro end-to-end-routing?          empty
         +--ro boundary-rerouting?          empty
         +--ro segment-based-rerouting?     empty
         +--ro lsp-integrety-required?      empty
         +--ro contiguous-lsp?              empty
         +--ro lsp-stitching-desired?       empty
         +--ro preplanned-lsp?              empty
         +--ro oob-mapping?                 empty
   augment /ietf-te:te/ietf-te:interfaces/ietf-te:interface:
   augment /ietf-te:te/ietf-te:lsps-state/ietf-te:lsp:
      +--ro lsp-source?                  inet:ip-address
      +--ro lsp-signaled-name?           string
      +--ro lsp-priority-setup?          uint8
      +--ro lsp-priority-hold?           uint8
      +--ro local-recording-desired?     empty
      +--ro se-style-desired?            empty
      +--ro path-reevaluation-request?   empty
      +--ro soft-preemption-desired?     empty
      +--ro end-to-end-routing?          empty
      +--ro boundary-rerouting?          empty
      +--ro segment-based-rerouting?     empty
      +--ro lsp-integrety-required?      empty
      +--ro contiguous-lsp?              empty
      +--ro lsp-stitching-desired?       empty
      +--ro preplanned-lsp?              empty
      +--ro oob-mapping?                 empty
      +--ro incoming-explicit-route
      |  +--ro explicit-route-subobjects* [subobject-index]
      |     +--ro subobject-index     uint32
      |     +--ro (type)?
      |        +--:(ipv4-address)
      |        |  +--ro v4-address?         inet:ipv4-address
      |        |  +--ro v4-prefix-length?   uint8
      |        |  +--ro v4-loose?           boolean
      |        +--:(ipv6-address)
      |        |  +--ro v6-address?         inet:ipv6-address
      |        |  +--ro v6-prefix-length?   uint8
      |        |  +--ro v6-loose?           boolean
      |        +--:(as-number)
      |        |  +--ro as-number?          uint16
      |        +--:(unnumbered-link)
      |        |  +--ro router-id?          inet:ip-address
      |        |  +--ro interface-id?       uint32
      |        +--:(label)
      |           +--ro value?              uint32
      +--ro outgoing-explicit-route
```

```
      |  +--ro explicit-route-subobjects* [subobject-index]
      |     +--ro subobject-index     uint32
      |     +--ro (type)?
      |        +--:(ipv4-address)
      |        |  +--ro v4-address?         inet:ipv4-address
      |        |  +--ro v4-prefix-length?   uint8
      |        |  +--ro v4-loose?           boolean
      |        +--:(ipv6-address)
      |        |  +--ro v6-address?         inet:ipv6-address
      |        |  +--ro v6-prefix-length?   uint8
      |        |  +--ro v6-loose?           boolean
      |        +--:(as-number)
      |        |  +--ro as-number?          uint16
      |        +--:(unnumbered-link)
      |        |  +--ro router-id?          inet:ip-address
      |        |  +--ro interface-id?       uint32
      |        +--:(label)
      |           +--ro value?              uint32
      +--ro path-record-route
      |  +--ro record-route-subobjects* [subobject-index]
      |     +--ro subobject-index     uint32
      |     +--ro (type)?
      |        +--:(ipv4-address)
      |        |  +--ro v4-address?         inet:ipv4-address
      |        |  +--ro v4-prefix-length?   uint8
      |        |  +--ro v4-flags?           uint8
      |        +--:(ipv6-address)
      |        |  +--ro v6-address?         inet:ipv6-address
      |        |  +--ro v6-prefix-length?   uint8
      |        |  +--ro v6-flags?           uint8
      |        +--:(label)
      |           +--ro value?              uint32
      |           +--ro flags?              uint8
      +--ro resv-record-route
         +--ro record-route-subobjects* [subobject-index]
            +--ro subobject-index     uint32
            +--ro (type)?
               +--:(ipv4-address)
               |  +--ro v4-address?         inet:ipv4-address
               |  +--ro v4-prefix-length?   uint8
               |  +--ro v4-flags?           uint8
               +--:(ipv6-address)
               |  +--ro v6-address?         inet:ipv6-address
               |  +--ro v6-prefix-length?   uint8
               |  +--ro v6-flags?           uint8
               +--:(label)
                  +--ro value?              uint32
                  +--ro flags?              uint8
```

5.2. RPC and Notification Data

TBD.

5.3. YANG Module

```
<CODE BEGINS> file "ietf-rsvp-te@2015-07-06.yang"
module ietf-rsvp-te {

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp-te";

  prefix "rsvp-te";

  import ietf-rsvp {
    prefix rsvp;
  }

  import ietf-te {
    prefix ietf-te;
  }

  import ietf-inet-types {
    prefix inet;
  }

  /* Import TE generic types */
  import ietf-te-types {
    prefix ietf-te-types;
  }

  organization
    "IETF TEAS Working Group";

  contact "TBA";

  description
    "This module contains the RSVP YANG generic data model.";

  revision 2015-07-06 {
    description "Latest revision to RSVP-TE generic YANG module";
    reference "RFC3209";
  }

  /**
   * RSVP-TE LSPs groupings.
   */

  grouping lsp-record-route-information_state {
```

```
        description "recorded route information grouping";
        container path-record-route {
          when "../origin-type != 'ingress'" {
            description "Applicable on non-ingress LSPs only";
          }
          description "RSVP Path recorded route information";
          list record-route-subobjects {
            key "subobject-index";
            description "";
            leaf subobject-index {
              type uint32;
              description "RRO subobject index";
            }
            uses ietf-te-types:record-route-subobject;
          }
        }
        container resv-record-route {
          when "../origin-type != 'egress'" {
            description "Applicable on non-egress LSPs only";
          }
          description "RSVP Resv recorded route information";
          list record-route-subobjects {
            key "subobject-index";
            description "";
            leaf subobject-index {
              type uint32;
              description "RRO subobject index";
            }
            uses ietf-te-types:record-route-subobject;
          }
        }
      }

      grouping lsp-explicit-route-information_state {
        description "RSVP-TE LSP explicit-route information";
        container incoming-explicit-route {
          when "../origin-type != 'ingress'" {
            description "Applicable on non-ingress LSPs only";
          }
          description "Incoming explicit route information";
          list explicit-route-subobjects {
            key "subobject-index";
            description "";
            leaf subobject-index {
              type uint32;
              description "ERO subobject index";
            }
            uses ietf-te-types:explicit-route-subobject;
```

```
        }
      }
      container outgoing-explicit-route {
        when "../origin-type != 'egress'" {
          description "Applicable on non-egress LSPs only";
        }
        description "Outgoing explicit route information";
        list explicit-route-subobjects {
          key "subobject-index";
          description "";
          leaf subobject-index {
            type uint32;
            description "ERO subobject index";
          }
          uses ietf-te-types:explicit-route-subobject;
        }
      }
    }

    grouping lsp-attributes-flags_config {
      description
        "Configuration parameters relating to RSVP-TE LSP
        attribute flags";
      leaf end-to-end-routing {
        type empty;
        description
          "End-to-end routing desired";
        reference "RFC4920, RFC5420";
      }
      leaf boundary-rerouting {
        type empty;
        description
          "Boundary rerouting desired";
        reference "RFC4920, RFC5420";
      }
      leaf segment-based-rerouting {
        type empty;
        description
          "Segment-based rerouting desired";
        reference "RFC4920, RFC5420";
      }
      leaf lsp-integrety-required {
        type empty;
        description "LSP integrity desired";
        reference "RFC4875";
      }
      leaf contiguous-lsp {
        type empty;
```

```
          description "Contiguous LSP";
          reference "RFC5151";
        }
        leaf lsp-stitching-desired {
          type empty;
          description "Stiticed LSP";
          reference "RFC5150";
        }
        leaf preplanned-lsp {
          type empty;
          description "Preplanned LSP";
          reference "RFC6001";
        }
        leaf oob-mapping {
          type empty;
          description
            "Mapping is done out-of-band";
          reference "RFC6511";
        }
      }

      grouping lsp-session-attributes-obj-flags_config {
        description
          "Configuration parameters relating to RSVP-TE LSP
          session attribute flags";
        reference
          "RFC4859: Registry for RSVP-TE Session Flags";
        leaf local-recording-desired {
          type empty;
          description "Path recording is desired.";
          reference "RFC3209";
        }
        leaf se-style-desired {
          type empty;
          description "SE Style desired";
          reference "RFC3209";
        }
        leaf path-reevaluation-request {
          type empty;
          description "Path re-evaluation request";
          reference "RFC4736";
        }
        leaf soft-preemption-desired {
          type empty;
          description "Soft-preemption is desired";
          reference "RFC5712";
        }
      }
```

```
     grouping lsp-properties_config {
       description
         "Configuration parameters relating to RSVP-TE LSP
         session attribute flags";
       leaf lsp-source {
         type inet:ip-address;
         description
           "LSP source address.";
       }
       leaf lsp-signaled-name {
         type string;
         description
           "Sets the session name to use in the session
           attribute object.";
       }
       leaf lsp-priority-setup {
         type uint8 {
           range "0..7";
         }
         description
           "RSVP session attributes setup priority";
       }
       leaf lsp-priority-hold {
         type uint8 {
           range "0..7";
         }
         description
           "RSVP session attributes hold priority";
       }
       uses lsp-session-attributes-obj-flags_config;
       uses lsp-attributes-flags_config;
     }

     grouping tunnel-properties {
       description
         "Top level grouping for LSP properties.";
       container config {
         description
           "Configuration parameters relating to
           LSP properties";
         uses lsp-properties_config;
       }
       container state {
         config false;
         description
           "State information associated with LSP
           properties";
         uses lsp-properties_config;
```

```
      }
    }
    /*** End of RSVP-TE LSP groupings ***/

    /**
     * RSVP-TE generic global properties.
     */

    grouping global-soft-preemption_config {
      description
        "Configuration for global RSVP-TE soft preemption";
      leaf soft-preemption-timeout {
        type uint16 {
          range 0..300;
        }
        default 0;
        description
          "Timeout value for soft preemption to revert
          to hard preemption";
      }
    }

    grouping global-soft-preemption {
      description
        "Top level group for RSVP-TE soft-preemption";
      container global-soft-preemption {
        presence "Enables soft preemption on a node.";
        description
          "Top level container for RSVP-TE soft-preemption";
        container config {
          description
            "Configuration parameters relating to RSVP
            soft preemption support";
          uses global-soft-preemption_config;
        }
        container state {
          description "State parameters relating to RSVP
            soft preemption support";
          uses global-soft-preemption_config;
        }
      }
    }
    /*** End of RSVP-TE generic global properties. ***/

    /**
     * RSVP-TE interface generic groupings.
     */
```

```
     grouping rsvp-te-interface-attributes {
       description
         "Top level grouping for RSVP-TE interface properties.";
       container rsvp-te-interface-attributes {
         description
           "Top level container for RSVP-TE interface
           properties";
         container config {
           description
             "Configuration parameters relating to RSVP-TE
             bandwidth";
         }
         container state {
           config false;
           description
             "State information associated with RSVP-TE
             bandwidth";
         }
       }
     }
     /*** End of RSVP-TE generic groupings ***/


     /* RSVP-TE global properties */
     augment "/rsvp:rsvp/rsvp:globals" {
       description
         "RSVP-TE augmentation to RSVP globals";
     }

     /* Linkage to the base RSVP all links */
     augment "/rsvp:rsvp/rsvp:interfaces" {
       description
         "RSVP-TE generic data augmentation pertaining to interfaces";
       /* To be added */
     }

     /* Linkage to per RSVP interface */
     augment "/rsvp:rsvp/rsvp:interfaces/rsvp:interface" {
       description
         "RSVP-TE generic data augmentation pertaining to specific
          interface";
       /* To be added */
     }

     /* add augmentation for sessions neighbors */
     augment "/rsvp:rsvp/rsvp:sessions" {
       description
         "RSVP-TE generic data augmentation pertaining to session";
```

```
      /* To be added */
    }

    augment "/rsvp:rsvp/rsvp:neighbors" {
      description
        "RSVP-TE generic data augmentation pertaining to neighbors";
      /* To be added */
    }

    /**
     * RSVP-TE generic augmentations of generic TE model.
     */

    augment "/ietf-te:te/ietf-te:tunnels/ietf-te:tunnel" {
      description
        "RSVP-TE generic data augmentation pertaining to TE tunnels";
      uses tunnel-properties;
    }

    augment "/ietf-te:te/ietf-te:interfaces/ietf-te:interface" {
      description
        "RSVP-TE generic data augmentation pertaining to specific TE
         interface";
    }

    augment "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp" {
      description
        "RSVP-TE generic data augmentation pertaining to specific TE
         LSP";
      uses lsp-properties_config;
      uses lsp-explicit-route-information_state;
      uses lsp-record-route-information_state;
    }
  }
  <CODE ENDS>
```

6. RSVP-TE MPLS Packet Model

This section describes the MPLS packet RSVP-TE YANG module that augments the RSVP-TE generic module to signal packet MPLS LSPs. RSVP-TE YANG modules for other dataplane technologies (e.g. OTN or WDM) will be defined in separate modules and in other drafts.

The following subsections describe the configuration and state data . pertaining to RSVP-TE packet MPLS YANG data model.

6.1. Configuration and State Data

The following are possible types of configuration and state data nodes in this module:

- those augmenting or extending the generic RSVP-TE module

- those augmenting or extending the base TE generic module

- those that are specific to the RSVP-TE packet module

Below is a YANG tree representation for data items defined in the RSVP-TE packet MPLS module:

```
module: ietf-rsvp-te-psc
augment /rsvp:rsvp/rsvp:globals:
   +--rw config
   |  +--rw rsvp-frr-local-revert!
   |     +--rw rsvp-frr-local-revert-delay?   uint32
   +--ro state
      +--ro rsvp-frr-local-revert!
         +--ro rsvp-frr-local-revert-delay?   uint32
augment /rsvp:rsvp/rsvp:interfaces:
augment /rsvp:rsvp/rsvp:interfaces/rsvp:interface:
augment /rsvp:rsvp/rsvp:sessions:
augment /rsvp:rsvp/rsvp:neighbors:
augment /ietf-te:te/ietf-te:tunnels/ietf-te:tunnel:
   +--rw config
   |  +--rw local-protection-desired?       empty
   |  +--rw bandwidth-protection-desired?   empty
   |  +--rw node-protection-desired?        empty
   |  +--rw non-php-desired?                empty
   |  +--rw entropy-label-cap?              empty
   |  +--rw oam-mep-entities-desired?       empty
   |  +--rw oam-mip-entities-desired?       empty
   +--ro state
      +--ro local-protection-desired?       empty
      +--ro bandwidth-protection-desired?   empty
      +--ro node-protection-desired?        empty
      +--ro non-php-desired?                empty
      +--ro entropy-label-cap?              empty
      +--ro oam-mep-entities-desired?       empty
      +--ro oam-mip-entities-desired?       empty
augment /ietf-te:te/ietf-te:interfaces/ietf-te:interface:
   +--rw config
   |  +--rw (bandwidth-value)?
   |  |  +--:(absolute)
   |  |  |  +--rw absolute-value?   uint32
```

```
      |  |  +--:(precentage)
      |  |     +--rw percent-value?    uint32
      |  +--rw (bc-model-type)?
      |     +--:(bc-model-rdm)
      |     |  +--rw bc-model-rdm
      |     |     +--rw bandwidth-psc-constraints
      |     |        +--rw maximum-reservable?   uint32
      |     |        +--rw bc-value*             uint32
      |     +--:(bc-model-mam)
      |     |  +--rw bc-model-mam
      |     |     +--rw bandwidth-psc-constraints
      |     |        +--rw maximum-reservable?   uint32
      |     |        +--rw bc-value*             uint32
      |     +--:(bc-model-mar)
      |        +--rw bc-model-mar
      |           +--rw bandwidth-psc-constraints
      |              +--rw maximum-reservable?   uint32
      |              +--rw bc-value*             uint32
      +--ro state
      |  +--ro (bandwidth-value)?
      |  |  +--:(absolute)
      |  |  |  +--ro absolute-value?                  uint32
      |  |  +--:(precentage)
      |  |     +--ro percent-value?                   uint32
      |  +--ro (bc-model-type)?
      |  |  +--:(bc-model-rdm)
      |  |  |  +--ro bc-model-rdm
      |  |  |     +--ro bandwidth-psc-constraints
      |  |  |        +--ro maximum-reservable?   uint32
      |  |  |        +--ro bc-value*             uint32
      |  |  +--:(bc-model-mam)
      |  |  |  +--ro bc-model-mam
      |  |  |     +--ro bandwidth-psc-constraints
      |  |  |        +--ro maximum-reservable?   uint32
      |  |  |        +--ro bc-value*             uint32
      |  |  +--:(bc-model-mar)
      |  |     +--ro bc-model-mar
      |  |        +--ro bandwidth-psc-constraints
      |  |           +--ro maximum-reservable?   uint32
      |  |           +--ro bc-value*             uint32
      |  +--ro interface-softpreemption-state
      |  |  +--ro soft-preempted-bandwidth?   uint32
      |  |  +--ro lsps* [source destination tunnel-id lsp-id
                         extended-tunnel-id]
      |  |     +--ro source                leafref
      |  |     +--ro destination           leafref
      |  |     +--ro tunnel-id             leafref
      |  |     +--ro lsp-id                leafref
```

```
      |  |     +--ro extended-tunnel-id    leafref
      |  |     +--ro type?                 leafref
      |  +--ro over-subscribed-bandwidth?        uint32
      +--rw rsvp-te-frr-backups {ietf-te-types:frr-te}?
         +--rw config
         |  +--rw backup-bandwidth?               uint32
         |  +--rw backup-bandwidth-classtype?     uint32
         |  +--rw (type)?
         |     +--:(static-tunnel)
         |     |  +--rw static-backups* [tunnel-name]
         |     |     +--rw tunnel-name    string
         |     +--:(auto-tunnel)
         |        +--rw auto-backup-protection?         identityref
         |        +--rw auto-backup-path-computation?   identityref
         +--ro state
            +--ro backup-bandwidth?               uint32
            +--ro backup-bandwidth-classtype?     uint32
            +--ro (type)?
               +--:(static-tunnel)
               |  +--ro static-backups* [tunnel-name]
               |     +--ro tunnel-name    string
               +--:(auto-tunnel)
                  +--ro auto-backup-protection?         identityref
                  +--ro auto-backup-path-computation?   identityref
   augment /ietf-te:te/ietf-te:lsps-state/ietf-te:lsp:
      +--ro local-protection-desired?       empty
      +--ro bandwidth-protection-desired?   empty
      +--ro node-protection-desired?        empty
      +--ro non-php-desired?                empty
      +--ro entropy-label-cap?              empty
      +--ro oam-mep-entities-desired?       empty
      +--ro oam-mip-entities-desired?       empty
      +--ro backup-info
         +--ro backup-tunnel-name?         string
         +--ro backup-frr-on?              uint8
         +--ro backup-protected-lsp-num?   uint32
```

6.2. RPC and Notification Data

TBD.

6.3. YANG Module

```
<CODE BEGINS> file "ietf-rsvp-te-psc@2015-07-06.yang"
module ietf-rsvp-te-psc {

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp-te-psc";
```

```
     prefix "rsvp-te-psc";

     import ietf-rsvp {
       prefix rsvp;
     }

     import ietf-te {
       prefix ietf-te;
     }

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

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

     organization
       "IETF TEAS Working Group";

     contact "TBA";

     description
       "Latest update to RSVP-TE packet YANG data model.";

     revision 2015-07-06 {
       description "Update to RSVP-TE packet YANG initial revision.";
       reference "RFC3209, RFC6511, RFC6790, RFC7260, RFC4859, RFC4090";
     }

     /* RSVP-TE LSPs packet groupings */
     grouping lsp-attributes-flags-psc_config {
       description
         "Configuration parameters relating to RSVP-TE LSP
         packet attribute flags";
       leaf non-php-desired {
         type empty;
         description
           "Non-PHP is desired";
         reference "RFC6511";
       }
       leaf entropy-label-cap {
         type empty;
         description "Entropy label capability";
         reference "RFC6790";
       }
       leaf oam-mep-entities-desired {
```

```
        type empty;
        description "OAM MEP entities desired";
        reference "RFC7260";
      }
      leaf oam-mip-entities-desired {
        type empty;
        description "OAM MIP entities desired";
        reference "RFC7260";
      }
    }

    grouping lsp-session-attributes-obj-flags-psc_config {
      description
        "Configuration parameters relating to RSVP-TE LSP
        packet session attribute flags";
      reference
        "RFC4859: Registry for RSVP-TE Session Flags";
      leaf local-protection-desired {
        type empty;
        description "Fastreroute local protection is desired.";
        reference
          "RFC4859: Registry for RSVP-TE Session Flags";
      }
      leaf bandwidth-protection-desired {
        type empty;
        description
          "Request FRR bandwidth protection on LSRs if
          present.";
        reference "RFC4090";
      }
      leaf node-protection-desired {
        type empty;
        description
          "Request FRR node protection on LSRs if
          present.";
        reference "RFC4090";
      }
    }

    grouping lsp-properties-psc {
      description
        "Top level grouping for LSP properties.";
      container config {
        description
          "Configuration parameters relating to
          LSP properties";
        uses lsp-session-attributes-obj-flags-psc_config;
        uses lsp-attributes-flags-psc_config;
```

```
        }
        container state {
          config false;
          description
            "State information associated with LSP
            properties";
          uses lsp-session-attributes-obj-flags-psc_config;
          uses lsp-attributes-flags-psc_config;
        }
      }
      /* End of RSVP-TE LSPs packet groupings */

      /* RSVP-TE packet interface groupings */
      grouping rsvp-te-interface_state {
        description
          "The RSVP-TE interface state grouping";
        leaf over-subscribed-bandwidth {
          type uint32;
          description
            "The amount of over-subscribed bandwidth on
            the interface";
        }
      }

      grouping rsvp-te-interface-softpreemption_state {
        description
          "The RSVP-TE interface preeemptions state grouping";
        container interface-softpreemption-state {
          description
            "The RSVP-TE interface preeemptions state grouping";
          leaf soft-preempted-bandwidth {
            type uint32;
            description
              "The amount of soft-preempted bandwidth on
              this interface";
          }
          list lsps {
            key
              "source destination tunnel-id lsp-id "+
              "extended-tunnel-id";
            description
              "List of LSPs that are soft-preempted";
            leaf source {
              type leafref {
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                     "ietf-te:source";
              }
              description
```

```
                "Tunnel sender address extracted from
                SENDER_TEMPLATE  object";
              reference "RFC3209";
            }
            leaf destination {
              type leafref {
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                     "ietf-te:destination";
              }
              description
                "Tunnel endpoint address extracted from
                SESSION object";
              reference "RFC3209";
            }
            leaf tunnel-id {
              type leafref {
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                  "ietf-te:tunnel-id";
              }
              description
                "Tunnel identifier used in the SESSION
                that remains constant over the life
                of the tunnel.";
              reference "RFC3209";
            }
            leaf lsp-id {
              type leafref {
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                     "ietf-te:lsp-id";
              }
              description
                "Identifier used in the SENDER_TEMPLATE
                and the FILTER_SPEC that can be changed
                to allow a sender to share resources with
                itself.";
              reference "RFC3209";
            }
            leaf extended-tunnel-id {
              type leafref {
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                  "ietf-te:extended-tunnel-id";
              }
              description
                "Extended Tunnel ID of the LSP.";
              reference "RFC3209";
            }
            leaf type {
              type leafref {
```

```
                path "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp/"+
                  "ietf-te:type";
              }
              description "LSP type P2P or P2MP";
            }
          }
        }
      }
      /* End of RSVP-TE interface groupings */

      /* RSVP-TE FRR groupings */
      grouping rsvp-te-frr-backups_config {
        description
          "Top level container for RSVP-TE FRR backup parameters";
        leaf backup-bandwidth {
          type uint32;
          description
            "Maximum bandwidth this facility backup
            is allowed to protect";
        }
        leaf backup-bandwidth-classtype {
          type uint32;
          description
            "Type of primary LSP bandwidth that the
            backup is allowed to protect.";
        }
        choice type {
          description
            "FRR backup tunnel type";
          case static-tunnel {
            list static-backups {
              key "tunnel-name";
              description
                "List of static backup tunnels that
                protect the RSVP-TE interface.";
              leaf tunnel-name {
                type string;
                description "FRR Backup tunnel";
              }
            }
          }
          case auto-tunnel {
            leaf auto-backup-protection {
              type identityref {
                base ietf-te-psc-types:backup-protection-type;
              }
              default
                ietf-te-psc-types:backup-protection-node-link;
```

```
            description
              "Describes whether the backup should offer
              protection against link, node, or either";
          }
          leaf auto-backup-path-computation {
            type identityref {
              base
                ietf-te-types:path-computation-srlg-type;
            }
            description
              "FRR backup computation type";
          }
        }
      }
    }

    grouping rsvp-te-frr-backups {
      description
        "Top level grouping for RSVP-TE FRR backup properties.";
      container rsvp-te-frr-backups {
        if-feature ietf-te-types:frr-te;
        description
          "Top level container for RSVP-TE FRR backup
          properties.";
        container config {
          description
            "Configuration parameters for interface RSVP-TE
            FRR backup properties";
          uses rsvp-te-frr-backups_config;
        }
        container state {
          config false;
          description
            "State parameters for interface RSVP-TE
            FRR backup properties";
          uses rsvp-te-frr-backups_config;
        }
      }
    }

    grouping lps-backup-info_state {
      description "Backup/bypass LSP related information";
      container backup-info {
        description
          "backup information";

        leaf backup-tunnel-name {
          type string;
```

```
          description
            "If an LSP has an FRR backup LSP that can protect it,
            this field identifies the tunnel name of the backup LSP.
            Otherwise, this field is empty.";
        }

        leaf backup-frr-on {
          type uint8;
          description
            "Whether currently this backup is carrying traffic";
        }

        leaf backup-protected-lsp-num {
          type uint32;
          description
            "Number of LSPs protected by this backup";
        }
      }
    }

    grouping rsvp-frr-local-revert_config {
      description "RSVP-TE FRR local revertive grouping";
      container rsvp-frr-local-revert {
        presence "Enable RSVP FRR local revertive recovery
          mode.";
        description
          "RSVP-TE global properties container";
        leaf rsvp-frr-local-revert-delay {
          type uint32;
          description
            "Time to wait after primary link is restored
            before node attempts local revertive
            procedures.";
        }
      }
    }

    /*** End of RSVP-TE FRR backup information ***/

    grouping globals-properties {
      description
        "Top level grouping for globals properties";
      container config {
        description
          "Configuration parameters relating to
           global RVP-TE packet properties";
        uses rsvp-frr-local-revert_config;
      }
```

```
      container state {
        config false;
        description
          "State parameters relating to
           global RVP-TE packet properties";
        uses rsvp-frr-local-revert_config;
      }
    }

    /* RSVP-TE global propeerties */
    augment "/rsvp:rsvp/rsvp:globals" {
      description
        "RSVP-TE augmentation to RSVP globals";
      uses globals-properties;
    }

    grouping rsvp-te-interface-attributes-psc {
      description
        "Top level grouping for RSVP-TE packet interface
        properties.";
      container config {
        description
          "Configuration parameters relating to RSVP-TE
          bandwidth";
        uses ietf-te-psc-types:bandwidth-psc-reservable;
      }
      container state {
        config false;
        description
          "State information associated with RSVP-TE
          bandwidth";
        uses ietf-te-psc-types:bandwidth-psc-reservable;
        uses rsvp-te-interface-softpreemption_state;
        uses rsvp-te-interface_state;
      }
    }

    /* Linkage to the base RSVP all links */
    augment "/rsvp:rsvp/rsvp:interfaces" {
      description "TBD";
      /* To be added */
    }

    /* Linkage to per RSVP link */
    augment "/rsvp:rsvp/rsvp:interfaces/rsvp:interface" {
      description "TBD";
      /* To be added */
    }
```

```
     /* add augmentation for sessions neighbors */
     augment "/rsvp:rsvp/rsvp:sessions" {
       description "TBD";
       /* To be added */
     }

     augment "/rsvp:rsvp/rsvp:neighbors" {
       description "TBD";
       /* To be added */
     }

     /**
      * Augmentation to TE generic module
      */
     augment "/ietf-te:te/ietf-te:tunnels/ietf-te:tunnel" {
       description "TBD";
       uses lsp-properties-psc;
     }

     augment "/ietf-te:te/ietf-te:interfaces/ietf-te:interface" {
       description
         "RSVP reservable bandwidth configuration properties";
       uses rsvp-te-interface-attributes-psc;
       uses rsvp-te-frr-backups;
     }

     augment "/ietf-te:te/ietf-te:lsps-state/ietf-te:lsp" {
       description
         "RSVP-TE LSP state properties";
       uses lsp-session-attributes-obj-flags-psc_config;
       uses lsp-attributes-flags-psc_config;
       uses lps-backup-info_state;
     }
   }
   <CODE ENDS>
```

7. 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 XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-rsvp-te 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-rsvp namespace: urn:ietf:params:xml:ns:yang:ietf-rsvp prefix: ietf-rsvp reference: RFC3209

name: ietf-rsvp-te namespace: urn:ietf:params:xml:ns:yang:ietf-rsvp-te prefix: ietf-rsvp-te reference: RFC3209

8. 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.

9. Acknowledgement

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

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-19 (work in progress), May 2015.

[I-D.saad-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Chen, X., and R. Jones, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-saad-teas-yang-te-01 (work in progress), March 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, January 2000.

[RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC 2961, April 2001.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC5063] Satyanarayana, A. and R. Rahman, "Extensions to GMPLS Resource Reservation Protocol (RSVP) Graceful Restart", RFC 5063, October 2007.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

10.2. Informative References

[I-D.openconfig-mpls-consolidated-model]
George, J., Fang, L., eric.osborne@level3.com, e., and R. Shakir, "MPLS / TE Model for Service Provider Networks", draft-openconfig-mpls-consolidated-model-00 (work in progress), March 2015.

[I-D.openconfig-netmod-opstate]
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-00 (work in progress), March 2015.

Authors' Addresses

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc

Email: tsaad@cisco.com

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Xufeng Liu
Ericsson

Email: xufeng.liu@ericsson.com

Himanshu Shah
Ciena

Email: tsaad@cisco.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

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2016

T. Saad, Ed.
R. Gandhi
Cisco Systems Inc
X. Liu
Ericsson
V. Beeram
Juniper Networks
H. Shah
Ciena
X. Chen
Huawei Technologies
R. Jones
Brocade
B. Wen
Comcast

July 06, 2015

A YANG Data Model for Traffic Engineering Tunnels and Interfaces
draft-saad-teas-yang-te-02

Abstract

This document defines a YANG data model for the configuration and management of Traffic Engineering (TE) interfaces and tunnels. The model defines generic data that is reusable across multiple data and control plane protocols.

The data model covers the configuration, operational state, remote procedural calls, and event notifications data for TE data.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g. ReST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage TE interfaces and P2P or P2MP TE tunnels. This data model restricts to TE generic data that is control and data plane agnostic. It is expected that other protocol and data plane specific modules (e.g. RSVP-TE [RFC3209]) will augment this TE model.

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].

1.2. Tree Diagram

A simplified graphical representation of the data model is presented in each section of the model. The following notations are used for the YANG model data tree representation.

```
<status> <flags> <name> <opts> <type>

 <status> is one of:
   +  for current
   x  for deprecated
   o  for obsolete

 <flags> is one of:
   rw  for read-write configuration data
   ro  for read-only non-configuration data
   -x  for execution rpcs
   -n  for notifications

 <name> is the name of the node

If the node is augmented into the tree from another module, its name
is printed as <prefix>:<name>

 <opts> is one of:
   ? for an optional leaf or node
   ! for a presence container
   * for a leaf-list or list
   Brackets [<keys>] for a list's keys
   Curly braces {<condition>} for optional feature that make node
conditional
   Colon : for marking case nodes
   Ellipses ("...") subtree contents not shown

   Parentheses enclose choice and case nodes, and case nodes are also
   marked with a colon (":").

 <type> is the name of the type for leafs and leaf-lists.
```

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]

Table 1: Prefixes and corresponding YANG modules

1.4. Open Issues and Next Steps

This document describes the YANG data model for the TE generic and helper modules. It also describes the high-level relationship between these modules and to other external protocol modules. The current revision of the draft focuses on configuration and state data aspects of the model. It is expected that the future revisions will cover RPC, and notification aspects.

Also, the models that define technology specific extensions to the generic TE model (e.g. OTN [RFC4328] TE extensions), are expected to be addressed in separate documents.

1.4.1. State Data Organization

Pure state data (for example, ephemeral or protocol derived state objects) can be modeled using one of the options below:

- o Contained inside the read-write container, under the "state" sub-container, as shown in Figure 3

- o Contained inside a separate read-only container, for example a tunnels-state container

The first option allows for the reusing of the containers that hold configuration data (in the "config" sub-container), and by placing state data under the read-only "state" sub-container of the parent container. However, when adopting this approach for ephemeral or purely derived states (e.g. auto tunnels), and since in this case the state hangs off the read-write parent container, it will be possible to delete the parent container and subsequently the ephemeral read-only state contained within (see Figure 3).

The second option entails defining a new read-only parent container in the model (e.g. neighbors-state) that holds the data.

This revision of the draft adopts the first option. Further discussions on this topic are expected to close on the best choice to adopt.

2. Data Model Overview

Although the basis of TE elements remain similar across different vendor implementations, however, the details of a TE model will usually vary across different vendor implementations. Also, implementations may vary in their support of the complete set of TE features. The TE YANG module defined in this document is an attempts to define a vendor agnostic model that will prescribe to IETF

standard terminology when different representation of data is possible.

The model is composed of common building blocks that are independent of specific data or control plane instantiations. It covers data representation for the configuration, state, remote procedural calls (RPCs), and event notifications.

Throughout the model, the approach described in [I-D.openconfig-netmod-opstate] is adopted to represent data pertaining to configuration intended state, applied state and derived state data elements. Each container in the model hold 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.

The decision to use this approach was made to better align with the MPLS consolidated model in [I-D.openconfig-mpls-consolidated-model], and maximize reusability of groupings defined in this document and allow for possible convergence between the two models.

2.1. Design Objectives

The goal of this document is to define a TE data model that can represent different TE vendor implementations, while adhering to standard terminology and behavior when resolving differences in implementations.

The following considerations with respect data organization are taken into account when defining the model:

- o reusable data elements are grouped into separate TE types module(s) that can be readily imported by other modules whenever needed

- o reusable TE data types that are data plane independent are grouped in the TE generic types module "ietf-te-types.yang"

- o reusable TE data elements that are data plane specific (e.g. packet PSC or switching technologies as defined in [RFC3473]) are expected to be grouped in a technology- specific types module, e.g. "ietf-te-psc-types.yang". It is expected that technology specific types will augment TE generic types as shown in Figure 1

```
            +---------+
            | ietf-te |       ^: import
            +---------+       o: augment
          import ^
                 |
                 |
           +---------------+
           | ietf-te-types |
           +---------------+
              o          o
              |            \
              |             \
       +-------------------+  +-------------------+
       | ietf-te-psc-types |  | ietf-te-otn-types |
       +-------------------+  +-------------------+
                               (shown for illustration
                                not in this document)
```

Figure 1: Relationship between generic and technology specific TE types modules

- o TE generic module includes data elements that are control plane independent. Data elements specific to a control plane protocol (e.g. RSVP-TE [RFC3209]) are expected to be in a separate module that augments the TE generic module. It is also expected that data relevant to a specific instantiations of data plane technology will exist in a separate YANG module that augments the TE generic model, see Figure 2.

```
     TE generic     +---------+        ^: import
     module         | ietf-te |        o: augment
                    +---------+
                       |   o
                       |   |
                       v   |
                    +--------------+
     RSVP-TE module | ietf-rsvp-te |o . . .
                    +--------------+         \
                       ^   |                  \
                       |   o              +--------------------+
                    +-----------+         | ietf-rsvp-otn-te  |
     RSVP module    | ietf-rsvp |         +--------------------+
                    +-----------+            RSVP-TE with OTN
                                             extensions
                                            (shown for illustration
                                             not in this document)
```

Figure 2: Relationship of TE module with other control plane protocol modules

o In general, little information in the model is designated as "mandatory", to allow freedom to vendors to adapt the data model to their specific product implementation.

2.2. Optional Features

Optional features are features beyond the generic TE model, and hence, it is up to a vendor to decide whether or not to support of a particular feature on a particular device.

This module declares a number of TE functions as features (such as P2MP-TE, soft-preemption etc.). It is intended that vendors will extend this features list.

2.3. Configuration Inheritance

The defined data model supports configuration inheritance for tunnels, paths, and interfaces. Data elements defined in the main container (e.g. that encompasses the list of tunnels, interfaces, or paths) are assumed to apply equally to all elements of the list, unless overridden explicitly for a certain element (e.g. tunnel, interface or path). Vendors are expected to augment the above container(s) to provide the list of inheritance command for their implementations.

2.4. Vendor Configuration Models

There two main popular types of routing protocol configuration that vendors may support:

- o protocol centric - all the protocol related configuration is contained within the protocol itself. Configuration belonging to multiple instances of the protocol running in different routing-instances (e.g. VRFs) are contained under the default routing instance [I-D.ietf-netmod-routing-cfg]:

- o VRF centric - all the protocol related configuration for a routing- instance is contained within this routing-instance.

On-going discussions within the IETF community have converged on adopting the VRF centric approach. The proposed model in this document adheres to this conclusion.

3. TE Generic Model Organization

This model covers configuration, state, RPC, and notifications data pertaining to TE global parameters, interfaces, and tunnels parameters.

The container "te" is the top level container in this data model. The presence of this container is expected to enable TE function system wide.

The approach described in [I-D.openconfig-netmod-opstate] allows for modeling the intended and respective applied and derived state. The TE state data in this model falls into one of the following categories:

- o State corresponding to applied configuration

- o State corresponding to derived state, counters, stats, etc.

- o State corresponding to ephemeral data (e.g. LSPs, auto-tunnels, etc.)

Data for the first two categories are contained under the respective "state" sub-container of the intended object (e.g. tunnel). The last category falls under a separate - e.g. lsps-state- container that contains the attributes of a purely derived state data (e.g. ephemeral objects) that are not associated with any configuration as shown in Figure 3.

```
module: ietf-te
   +--rw te!
      +--rw globals
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--rw interfaces
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--rw tunnels
            +-- rw config
               <<intended configuration>>
            .
            +-- ro state
               <<applied configuration>>
               <<derived state associated with the tunnel>>
         .
         .
      +--ro tunnels-state
           <<ephemeral tunnels>>
rpcs:
   +---x globals-rpc
   +---x interfaces-rpc
   +---x tunnels-rpc
notifications:
   +---n globals-notif
   +---n interfaces-notif
   +---n tunnels-notif
```

Figure 3: TE highlevel model view

3.1. Global Configuration and State Data

This branch of the data model covers configurations that control TE features behavior system-wide, and its respective state. Examples of such configuration data are:

- o Table of named SRLG mappings
- o Table of named (extended) administrative groups mappings
- o Table of named explicit paths to be referenced by TE tunnels
- o Table of named path-constraints sets
- o Auto-bandwidth global parameters
- o TE diff-serve TE-class maps
- o System-wide capabilities for LSP reoptimization
  - * Reoptimization timers (periodic interval, LSP installation and cleanup)
- o System-wide capabilities for TE state flooding
  - * Periodic flooding interval
- o System-wide capabilities that affect the originating, traversing and terminating LSPs. For example:
  - * Path selection parameters (e.g. metric) at head-end LSR
  - * Path protection parameters at head-end LSR
  - * (Soft) preemption parameters
  - * Fast reroute parameters

The approach described in [I-D.openconfig-netmod-opstate] is utilised to include the global state data under the global "state" sub-container as shown in Figure 3.

Examples of such states are:

- o Global statistics (signaling, admission, preemption, flooding)
- o Global counters (number of tunnels/LSPs/interfaces)

```
module: ietf-te
   +--rw te!
      +--rw globals
      |  +--rw config
      |  +--ro state
      |  |  +--ro tunnels-counter?   uint32
```

```
         |  |  +--ro lsps-counter?      uint32
         |  +--rw named-admin-groups
         |  |  +--rw config
         |  |  |  +--rw named-admin-groups* [name]
         |  |  |     +--rw name     string
         |  |  |     +--rw group?   ietf-te-types:admin-groups
         |  |  +--ro state
         |  |     +--ro named-admin-groups* [name]
         |  |        +--ro name     string
         |  |        +--ro group?   ietf-te-types:admin-groups
         |  +--rw named-srlgs
         |  |  +--rw config
         |  |  |  +--rw named-srlgs* [name]
         |  |  |     +--rw name     string
         |  |  |     +--rw group?   ietf-te-types:srlg
         |  |  +--ro state
         |  |     +--ro named-srlgs* [name]
         |  |        +--ro name     string
         |  |        +--ro group?   ietf-te-types:srlg
         |  +--rw named-explicit-paths
         |  |  +--rw config
         |  |  |  +--rw named-explicit-paths* [name]
         |  |  |     +--rw name                     string
         |  |  |     +--rw explicit-route-objects* [index]
         |  |  |        +--rw index                  uint8
         |  |  |        +--rw (type)?
         |  |  |        |  +--:(ipv4-address)
         |  |  |        |  |  +--rw v4-address?      inet:ipv4-address
         |  |  |        |  |  +--rw v4-prefix-length?uint8
         |  |  |        |  |  +--rw v4-loose?        boolean
         |  |  |        |  +--:(ipv6-address)
         |  |  |        |  |  +--rw v6-address?      inet:ipv6-address
         |  |  |        |  |  +--rw v6-prefix-length?uint8
         |  |  |        |  |  +--rw v6-loose?        boolean
         |  |  |        |  +--:(as-number)
         |  |  |        |  |  +--rw as-number?       uint16
         |  |  |        |  +--:(unnumbered-link)
         |  |  |        |  |  +--rw router-id?       inet:ip-address
         |  |  |        |  |  +--rw interface-id?    uint32
         |  |  |        |  +--:(label)
         |  |  |        |     +--rw value?           uint32
         |  |  |        +--rw explicit-route-usage?   identityref
         |  |  +--ro state
         |  |     +--ro named-explicit-paths* [name]
         |  |        +--ro name                     string
         |  |        +--ro explicit-route-objects* [index]
         |  |           +--ro index                  uint8
         |  |           +--ro (type)?
```

```
          |  |           |  +--:(ipv4-address)
          |  |           |  |  +--ro v4-address?      inet:ipv4-address
          |  |           |  |  +--ro v4-prefix-length?uint8
          |  |           |  |  +--ro v4-loose?        boolean
          |  |           |  +--:(ipv6-address)
          |  |           |  |  +--ro v6-address?      inet:ipv6-address
          |  |           |  |  +--ro v6-prefix-length?uint8
          |  |           |  |  +--ro v6-loose?        boolean
          |  |           |  +--:(as-number)
          |  |           |  |  +--ro as-number?       uint16
          |  |           |  +--:(unnumbered-link)
          |  |           |  |  +--ro router-id?       inet:ip-address
          |  |           |  |  +--ro interface-id?    uint32
          |  |           |  +--:(label)
          |  |           |     +--ro value?           uint32
          |  |           +--ro explicit-route-usage?   identityref
          |  +--rw named-path-constraints
          |     +--rw config
          |     |  +--rw named-path-constraints* [name]
          |     |     +--rw name              string
          |     |     +--rw path-selection
          |     |        +--rw topology?                topology-id
          |     |        +--rw cost-limit?              uint32
          |     |        +--rw hop-limit?               uint8
          |     |        +--rw metric-type?             identityref
          |     |        +--rw tiebreaker-type?         identityref
          |     |        +--rw ignore-overload?         boolean
          |     |        +--rw tunnel-path-affinities
          |     |        |  +--rw (style)?
          |     |        |     +--:(values)
          |     |        |     |  +--rw value?         uint32
          |     |        |     |  +--rw mask?          uint32
          |     |        |     +--:(named)
          |     |        |        +--rw constraints* [usage]
          |     |        |           +--rw usage         identityref
          |     |        |           +--rw constraint
          |     |        |              +--rw affinity-names* [name]
          |     |        |                 +--rw name    string
          |     |        +--rw tunnel-path-srlgs
          |     |           +--rw (style)?
          |     |              +--:(values)
          |     |              |  +--rw usage?         identityref
          |     |              |  +--rw values*        srlg
          |     |              +--:(named)
          |     |                 +--rw constraints* [usage]
          |     |                    +--rw usage         identityref
          |     |                    +--rw constraint
          |     |                       +--rw srlg-names* [name]
```

```
|     |                          +--rw name    string
|     +--ro state
|        +--ro named-path-constraints* [name]
|           +--ro name              string
|           +--ro path-selection
|              +--ro topology?        topology-id
|              +--ro cost-limit?      uint32
|              +--ro hop-limit?       uint8
|              +--ro metric-type?     identityref
|              +--ro tiebreaker-type? identityref
|              +--ro ignore-overload? boolean
|              +--ro tunnel-path-affinities
|              |  +--ro (style)?
|              |     +--:(values)
|              |     |  +--ro value?         uint32
|              |     |  +--ro mask?          uint32
|              |     +--:(named)
|              |        +--ro constraints* [usage]
|              |           +--ro usage         identityref
|              |           +--ro constraint
|              |              +--ro affinity-names* [name]
|              |                 +--ro name    string
|              +--ro tunnel-path-srlgs
|                 +--ro (style)?
|                    +--:(values)
|                    |  +--ro usage?         identityref
|                    |  +--ro values*        srlg
|                    +--:(named)
|                       +--ro constraints* [usage]
|                          +--ro usage         identityref
|                          +--ro constraint
|                             +--ro srlg-names* [name]
|                                +--ro name    string
```

Figure 4: TE globals configuration and state tree

3.2. Interfaces Configuration and State Data

This branch of the data model covers configurations elements that control TE features behavior system-wide. Examples of such configuration data are:

This branch of the data model covers configurations that control TE features behavior system-wide, and its respective state. Examples of such configuration data are:

This branch of the model covers configuration and state data items, the corresponding applied state data, and possible derived state

pertaining to TE interfaces. Examples of tunnel configuration date for TE interfaces are:

- o Maximum reservable bandwidth, bandwidth constraints (BC)
- o Flooding parameters
  - * Flooding intervals and threshold values
- o Fast reroute backup tunnel properties (such as static, auto-tunnel)
- o interface attributes
  - * (Extended) administrative groups
  - * SRLG values
  - * TE metric value

```
module: ietf-te
   +--rw te!
      +--rw interfaces
      |  +--rw interface* [interface]
      |     +--rw interface                 if:interface-ref
      |     +--rw config
      |     |  +--rw te-metric?   ietf-te-types:te-metric
      |     +--ro state
      |     |  +--ro te-metric? ietf-te-types:te-metric
      |     |  +--ro interface-advertisements_state
      |     |     +--ro flood-interval?           uint32
      |     |     +--ro last-flooded-time?        uint32
      |     |     +--ro next-flooded-time?        uint32
      |     |     +--ro last-flooded-trigger?     enumeration
      |     |     +--ro advertized-level-areas* [level-area]
      |     |        +--ro level-area    uint32
      |     +--rw te-admin-groups
      |     |  +--rw config
      |     |  |  +--rw (admin-group-type)?
      |     |  |     +--:(value-admin-groups)
      |     |  |     |  +--rw (value-admin-group-type)?
      |     |  |     |     +--:(value-admin-groups)
      |     |  |     |     |  +--rw admin-group?
      |     |  |     |     +--:(value-extended-admin-groups)
      |     |  |     |        +--rw extended-admin-group?
      |     |  |     +--:(named-admin-groups)
      |     |  |        +--rw named-admin-groups* [named-admin-group]
      |     |  |           +--rw named-admin-group    leafref
```

```
        |     |  +--ro state
        |     |     +--ro (admin-group-type)?
        |     |        +--:(value-admin-groups)
        |     |        |  +--ro (value-admin-group-type)?
        |     |        |     +--:(value-admin-groups)
        |     |        |     |  +--ro admin-group?
        |     |        |     +--:(value-extended-admin-groups)
        |     |        |        +--ro extended-admin-group?
        |     |        +--:(named-admin-groups)
        |     |           +--ro named-admin-groups* [named-admin-group]
        |     |              +--ro named-admin-group    leafref
        |     +--rw te-srlgs
        |     |  +--rw config
        |     |  |  +--rw (srlg-type)?
        |     |  |     +--:(value-srlgs)
        |     |  |     |  +--rw values* [value]
        |     |  |     |     +--rw value    uint32
        |     |  |     +--:(named-srlgs)
        |     |  |        +--rw named-srlgs* [named-srlg]
        |     |  |           +--rw named-srlg    leafref
        |     |  +--ro state
        |     |     +--ro (srlg-type)?
        |     |        +--:(value-srlgs)
        |     |        |  +--ro values* [value]
        |     |        |     +--ro value    uint32
        |     |        +--:(named-srlgs)
        |     |           +--ro named-srlgs* [named-srlg]
        |     |              +--ro named-srlg    leafref
        |     +--rw te-switching-cap
        |     |  +--rw config
        |     |  |  +--rw switching-capabilities* [switching-capability]
        |     |  |     +--rw switching-capability    identityref
        |     |  |     +--rw encoding?               identityref
        |     |  +--ro state
        |     |     +--ro switching-capabilities* [switching-capability]
        |     |        +--ro switching-capability    identityref
        |     |        +--ro encoding?               identityref
        |     +--rw te-flooding-parameters
        |        +--rw config
        |        |  +--rw thresholds
        |        |     +--rw (type)?
        |        |        +--:(equal-steps)
        |        |        |  +--rw (equal-step-type)?
        |        |        |     +--:(up-down-different-step)
        |        |        |     |  +--rw up-step?      uint8
        |        |        |     |  +--rw down-step?    uint8
        |        |        |     +--:(up-down-same-step)
        |        |        |        +--rw step?         uint8
```

```
        |        |        +--:(unequal-steps)
        |        |           +--rw up-steps* [value]
        |        |           |  +--rw value    uint8
        |        |           +--rw down-steps* [value]
        |        |              +--rw value    uint8
        |        +--ro state
        |           +--ro thresholds
        |              +--ro (type)?
        |                 +--:(equal-steps)
        |                 |  +--ro (equal-step-type)?
        |                 |     +--:(up-down-different-step)
        |                 |     |  +--ro up-step?      uint8
        |                 |     |  +--ro down-step?    uint8
        |                 |     +--:(up-down-same-step)
        |                 |        +--ro step?         uint8
        |                 +--:(unequal-steps)
        |                    +--ro up-steps* [value]
        |                    |  +--ro value    uint8
        |                    +--ro down-steps* [value]
        |                       +--ro value    uint8
```

Figure 5: TE interfaces configuration and state tree

The state corresponding to the TE interfaces applied configuration, protocol derived state, and stats and counters all fall under the interface attributes "state" sub-container as shown in Figure 6 below:

```
module: ietf-te
   +--rw te!
      +--rw interfaces
          .
          +-- rw te-attributes
              +-- rw config
                 <<intended configuration>>
              .
              +-- ro state
                 <<applied configuration>>
                 <<derived state associated with the TE interface>>
```

Figure 6: TE interface state

This covers state data for TE interfaces such as:

- o Bandwidth information: maximum bandwidth, available bandwidth at different priorities and for each class-type (CT)

- o List of admitted LSPs

* Name, bandwidth value and pool, time, priority

o Statistics: state counters, flooding counters, admission counters (accepted/rejected), preemption counters

o Adjacency information

* Neighbor address

* Metric value

3.3. Tunnels Configuration and State Data

This branch of the model covers intended, and corresponding applied configuration for tunnels. As well, it holds possible derived state pertaining to TE tunnels.

The approach described in [I-D.openconfig-netmod-opstate] is utilised for the inclusion of operational and statistical data as shown in Figure 7.

```
module: ietf-te
   +--rw te!
      +--rw tunnels
         .
         +-- rw tunnel-properties
             +-- rw config
                <<intended configuration>>
             .
             +-- ro state
                <<applied configuration>>
                <<derived state associated with the tunnel>>
```

Figure 7: TE interface state tree

Examples of tunnel configuration date for TE tunnels:

o Name and type (e.g. P2P, P2MP) of tunnel

o Admin-state

o Primary and secondary paths

o Routing usage (auto-route announce, forwarding adjacency)

o Policy based routing (PBR) parameters

module: ietf-te

```
   +--rw te!
      +--rw tunnels
      |  +--rw tunnel* [name type]
      |     +--rw name          string
      |     +--rw type          identityref
      |     +--rw identifier?   uint16
      |     +--rw config
      |     |  +--rw description?            string
      |     |  +--rw admin-status?           identityref
      |     |  +--rw (routing-choice)?
      |     |  |  +--:(autoroute)
      |     |  |  |  +--rw autoroute-announce!
      |     |  |  |     +--rw routing-afs*       inet:ip-version
      |     |  |  |     +--rw (metric-type)?
      |     |  |  |        +--:(metric)
      |     |  |  |        |  +--rw metric?            uint32
      |     |  |  |        +--:(relative-metric)
      |     |  |  |        |  +--rw relative-metric?   int32
      |     |  |  |        +--:(absolute-metric)
      |     |  |  |           +--rw absolute-metric?   uint32
      |     |  |  +--:(forwarding-adjacency)
      |     |  |     +--rw forwarding-adjacency!
      |     |  |        +--rw holdtime?      uint32
      |     |  |        +--rw routing-afs*   inet:ip-version
      |     |  +--rw forwarding
      |     |  |  +--rw load-share?   uint32
      |     |  |  +--rw (policy-type)?
      |     |  |     +--:(class)
      |     |  |     |  +--rw class
      |     |  |     |     +--rw class?   uint8
      |     |  |     +--:(group)
      |     |  |        +--rw group
      |     |  |           +--rw classes*   uint8
      |     |  +--rw bidirectional
      |     |  |  +--rw association
      |     |  |     +--rw id?              uint16
      |     |  |     +--rw source?          inet:ip-address
      |     |  |     +--rw global-source?   inet:ip-address
      |     |  |     +--rw type?            identityref
      |     |  |     +--rw provisioing?     identityref
      |     |  +--rw (path-type)?
      |     |     +--:(p2p)
      |     |     |  +--rw destination?            inet:ip-address
      |     |     |  +--rw primary-paths* [preference]
      |     |     |     +--rw preference            uint8
      |     |     |     +--rw tunnel-path-params
      |     |     |     |  +--rw path-named-constraint?   leafref
      |     |     |     |  +--rw path-selection
```

```
      |     |     |     |  |  +--rw topology?         topology-id
      |     |     |     |  |  +--rw cost-limit?       uint32
      |     |     |     |  |  +--rw hop-limit?        uint8
      |     |     |     |  |  +--rw metric-type?      identityref
      |     |     |     |  |  +--rw tiebreaker-type?  identityref
      |     |     |     |  |  +--rw ignore-overload?  boolean
      |     |     |     |  |  +--rw tunnel-path-affinities
      |     |     |     |  |  |  +--rw (style)?
      |     |     |     |  |  |     +--:(values)
      |     |     |     |  |  |     |  +--rw value?         uint32
      |     |     |     |  |  |     |  +--rw mask?          uint32
      |     |     |     |  |  |     +--:(named)
      |     |     |     |  |  |        +--rw constraints* [usage]
      |     |     |     |  |  |          +--rw usage identityref
      |     |     |     |  |  |          +--rw constraint
      |     |     |     |  |  |            +--rw affinity-names* [name]
      |     |     |     |  |  |                +--rw name    string
      |     |     |     |  |  +--rw tunnel-path-srlgs
      |     |     |     |  |     +--rw (style)?
      |     |     |     |  |        +--:(values)
      |     |     |     |  |        |  +--rw usage?         identityref
      |     |     |     |  |        |  +--rw values*        srlg
      |     |     |     |  |        +--:(named)
      |     |     |     |  |           +--rw constraints* [usage]
      |     |     |     |  |              +--rw usage identityref
      |     |     |     |  |              +--rw constraint
      |     |     |     |  |                 +--rw srlg-names* [name]
      |     |     |     |  |                    +--rw name    string
      |     |     |     |  +--rw (type)?
      |     |     |     |  |  +--:(dynamic)
      |     |     |     |  |  |  +--rw dynamic?                empty
      |     |     |     |  |  +--:(explicit)
      |     |     |     |  |     +--rw explicit-path-name?      leafref
      |     |     |     |  +--rw no-cspf?                 empty
      |     |     |     |  +--rw lockdown?                empty
      |     |     |     +--rw seondary-paths* [preference]
      |     |     |        +--rw preference            uint8
      |     |     |        +--rw tunnel-path-params
      |     |     |           +--rw path-named-constraint?   leafref
      |     |     |           +--rw path-selection
      |     |     |           |  +--rw topology?        topology-id
      |     |     |           |  +--rw cost-limit?      uint32
      |     |     |           |  +--rw hop-limit?       uint8
      |     |     |           |  +--rw metric-type?     identityref
      |     |     |           |  +--rw tiebreaker-type? identityref
      |     |     |           |  +--rw ignore-overload? boolean
      |     |     |           |  +--rw tunnel-path-affinities
      |     |     |           |  |  +--rw (style)?
```

```
      |     |     |           |  |     +--:(values)
      |     |     |           |  |     |  +--rw value?         uint32
      |     |     |           |  |     |  +--rw mask?          uint32
      |     |     |           |  |     +--:(named)
      |     |     |           |  |       +--rw constraints* [usage]
      |     |     |           |  |        +--rw usage  identityref
      |     |     |           |  |        +--rw constraint
      |     |     |           |  |           +--rw affinity-names*[name]
      |     |     |           |  |              +--rw name    string
      |     |     |           |  +--rw tunnel-path-srlgs
      |     |     |           |     +--rw (style)?
      |     |     |           |        +--:(values)
      |     |     |           |        |  +--rw usage?  identityref
      |     |     |           |        |  +--rw values* srlg
      |     |     |           |        +--:(named)
      |     |     |           |           +--rw constraints* [usage]
      |     |     |           |              +--rw usage identityref
      |     |     |           |              +--rw constraint
      |     |     |           |                 +--rw srlg-names* [name]
      |     |     |           |                    +--rw name    string
      |     |     |           +--rw (type)?
      |     |     |           |  +--:(dynamic)
      |     |     |           |  |  +--rw dynamic?                empty
      |     |     |           |  +--:(explicit)
      |     |     |           |     +--rw explicit-path-name? leafref
      |     |     |           +--rw no-cspf?                empty
      |     |     |           +--rw lockdown?               empty
      |     |     +--:(p2mp) {ietf-te-types:p2mp-te}?
      |     |        +--rw p2mp-paths* [destination]
      |     |           +--rw destination      inet:ip-address
      |     |           +--rw primary-paths* [preference]
      |     |              +--rw preference            uint8
      |     |              +--rw tunnel-path-params
      |     |              |  +--rw path-named-constraint?   leafref
      |     |              |  +--rw path-selection
      |     |              |  |  +--rw topology?        topology-id
      |     |              |  |  +--rw cost-limit?      uint32
      |     |              |  |  +--rw hop-limit?       uint8
      |     |              |  |  +--rw metric-type?     identityref
      |     |              |  |  +--rw tiebreaker-type? identityref
      |     |              |  |  +--rw ignore-overload? boolean
      |     |              |  |  +--rw tunnel-path-affinities
      |     |              |  |  |  +--rw (style)?
      |     |              |  |  |     +--:(values)
      |     |              |  |  |     |  +--rw value?         uint32
      |     |              |  |  |     |  +--rw mask?          uint32
      |     |              |  |  |     +--:(named)
      |     |              |  |  |        +--rw constraints* [usage]
```

```
      |     |              |  |  |         +--rw usage identityref
      |     |              |  |  |         +--rw constraint
      |     |              |  |  |          +--rw affinity-names* [name]
      |     |              |  |  |                 +--rw name    string
      |     |              |  |  +--rw tunnel-path-srlgs
      |     |              |  |     +--rw (style)?
      |     |              |  |        +--:(values)
      |     |              |  |        |  +--rw usage? identityref
      |     |              |  |        |  +--rw values*        srlg
      |     |              |  |        +--:(named)
      |     |              |  |           +--rw constraints* [usage]
      |     |              |  |              +--rw usage identityref
      |     |              |  |              +--rw constraint
      |     |              |  |                 +--rw srlg-names* [name]
      |     |              |  |                    +--rw name    string
      |     |              |  +--rw (type)?
      |     |              |  |  +--:(dynamic)
      |     |              |  |  |  +--rw dynamic?                empty
      |     |              |  |  +--:(explicit)
      |     |              |  |     +--rw explicit-path-name? leafref
      |     |              |  +--rw no-cspf?                empty
      |     |              |  +--rw lockdown?               empty
      |     |              +--rw seondary-paths* [preference]
      |     |                 +--rw preference            uint8
      |     |                 +--rw tunnel-path-params
      |     |                    +--rw path-named-constraint?   leafref
      |     |                    +--rw path-selection
      |     |                    |  +--rw topology?        topology-id
      |     |                    |  +--rw cost-limit?      uint32
      |     |                    |  +--rw hop-limit?       uint8
      |     |                    |  +--rw metric-type?     identityref
      |     |                    |  +--rw tiebreaker-type? identityref
      |     |                    |  +--rw ignore-overload? boolean
      |     |                    |  +--rw tunnel-path-affinities
      |     |                    |  |  +--rw (style)?
      |     |                    |  |     +--:(values)
      |     |                    |  |     |  +--rw value?         uint32
      |     |                    |  |     |  +--rw mask?          uint32
      |     |                    |  |     +--:(named)
      |     |                    |  |      +--rw constraints* [usage]
      |     |                    |  |       +--rw usage identityref
      |     |                    |  |       +--rw constraint
      |     |                    |  |        +--rw affinity-names*[name]
      |     |                    |  |            +--rw name    string
      |     |                    |  +--rw tunnel-path-srlgs
      |     |                    |     +--rw (style)?
      |     |                    |        +--:(values)
      |     |                    |        |  +--rw usage? identityref
```

```
      |     |                   |        |  +--rw values*        srlg
      |     |                   |        +--:(named)
      |     |                   |           +--rw constraints* [usage]
      |     |                   |              +--rw usage identityref
      |     |                   |              +--rw constraint
      |     |                   |                 +--rw srlg-names*
      |     |                   |                    +--rw name string
      |     |                   +--rw (type)?
      |     |                   |  +--:(dynamic)
      |     |                   |  |  +--rw dynamic? empty
      |     |                   |  +--:(explicit)
      |     |                   |     +--rw explicit-path-name? leafref
      |     |                   +--rw no-cspf?                 empty
      |     |                   +--rw lockdown?                empty
      |     +--ro state
      |        +--ro description?            string
      |        +--ro admin-status?           identityref
      |        +--ro (routing-choice)?
      |        |  +--:(autoroute)
      |        |  |  +--ro autoroute-announce!
      |        |  |     +--ro routing-afs*       inet:ip-version
      |        |  |     +--ro (metric-type)?
      |        |  |        +--:(metric)
      |        |  |        |  +--ro metric?            uint32
      |        |  |        +--:(relative-metric)
      |        |  |        |  +--ro relative-metric?   int32
      |        |  |        +--:(absolute-metric)
      |        |  |           +--ro absolute-metric?   uint32
      |        |  +--:(forwarding-adjacency)
      |        |     +--ro forwarding-adjacency!
      |        |        +--ro holdtime?      uint32
      |        |        +--ro routing-afs*   inet:ip-version
      |        +--ro forwarding
      |        |  +--ro load-share?   uint32
      |        |  +--ro (policy-type)?
      |        |     +--:(class)
      |        |     |  +--ro class
      |        |     |     +--ro class?   uint8
      |        |     +--:(group)
      |        |        +--ro group
      |        |           +--ro classes*   uint8
      |        +--ro bidirectional
      |        |  +--ro association
      |        |     +--ro id?              uint16
      |        |     +--ro source?          inet:ip-address
      |        |     +--ro global-source?   inet:ip-address
      |        |     +--ro type?            identityref
      |        |     +--ro provisioing?     identityref
```

```
      |        +--ro (path-type)?
      |        |  +--:(p2p)
      |        |  |  +--ro destination?            inet:ip-address
      |        |  |  +--ro primary-paths* [preference]
      |        |  |     +--ro preference            uint8
      |        |  |     +--ro tunnel-path-params
      |        |  |     |  +--ro path-named-constraint?   leafref
      |        |  |     |  +--ro path-selection
      |        |  |     |  |  +--ro topology?        topology-id
      |        |  |     |  |  +--ro cost-limit?      uint32
      |        |  |     |  |  +--ro hop-limit?       uint8
      |        |  |     |  |  +--ro metric-type?     identityref
      |        |  |     |  |  +--ro tiebreaker-type? identityref
      |        |  |     |  |  +--ro ignore-overload? boolean
      |        |  |     |  |  +--ro tunnel-path-affinities
      |        |  |     |  |  |  +--ro (style)?
      |        |  |     |  |  |     +--:(values)
      |        |  |     |  |  |     |  +--ro value?         uint32
      |        |  |     |  |  |     |  +--ro mask?          uint32
      |        |  |     |  |  |     +--:(named)
      |        |  |     |  |  |        +--ro constraints* [usage]
      |        |  |     |  |  |           +--ro usage  identityref
      |        |  |     |  |  |           +--ro constraint
      |        |  |     |  |  |              +--ro affinity-names*[name]
      |        |  |     |  |  |                 +--ro name    string
      |        |  |     |  |  +--ro tunnel-path-srlgs
      |        |  |     |  |     +--ro (style)?
      |        |  |     |  |        +--:(values)
      |        |  |     |  |        |  +--ro usage?         identityref
      |        |  |     |  |        |  +--ro values*        srlg
      |        |  |     |  |        +--:(named)
      |        |  |     |  |           +--ro constraints* [usage]
      |        |  |     |  |              +--ro usage identityref
      |        |  |     |  |              +--ro constraint
      |        |  |     |  |                 +--ro srlg-names* [name]
      |        |  |     |  |                    +--ro name    string
      |        |  |     |  +--ro (type)?
      |        |  |     |  |  +--:(dynamic)
      |        |  |     |  |  |  +--ro dynamic?                 empty
      |        |  |     |  |  +--:(explicit)
      |        |  |     |  |     +--ro explicit-path-name?      leafref
      |        |  |     |  +--ro no-cspf?                empty
      |        |  |     |  +--ro lockdown?               empty
      |        |  |     +--ro seondary-paths* [preference]
      |        |  |        +--ro preference            uint8
      |        |  |        +--ro tunnel-path-params
      |        |  |           +--ro path-named-constraint?   leafref
      |        |  |           +--ro path-selection
```

```
      |        |  |          |  +--ro topology?        topology-id
      |        |  |          |  +--ro cost-limit?      uint32
      |        |  |          |  +--ro hop-limit?       uint8
      |        |  |          |  +--ro metric-type?     identityref
      |        |  |          |  +--ro tiebreaker-type? identityref
      |        |  |          |  +--ro ignore-overload? boolean
      |        |  |          |  +--ro tunnel-path-affinities
      |        |  |          |  |  +--ro (style)?
      |        |  |          |  |     +--:(values)
      |        |  |          |  |     |  +--ro value?         uint32
      |        |  |          |  |     |  +--ro mask?          uint32
      |        |  |          |  |     +--:(named)
      |        |  |          |  |        +--ro constraints* [usage]
      |        |  |          |  |           +--ro usage identityref
      |        |  |          |  |           +--ro constraint
      |        |  |          |  |              +--ro affinity-names*
      |        |  |          |  |                 +--ro name    string
      |        |  |          |  +--ro tunnel-path-srlgs
      |        |  |          |     +--ro (style)?
      |        |  |          |        +--:(values)
      |        |  |          |        |  +--ro usage? identityref
      |        |  |          |        |  +--ro values*        srlg
      |        |  |          |        +--:(named)
      |        |  |          |           +--ro constraints* [usage]
      |        |  |          |              +--ro usage identityref
      |        |  |          |              +--ro constraint
      |        |  |          |                 +--ro srlg-names* [name]
      |        |  |          |                    +--ro name    string
      |        |  |          +--ro (type)?
      |        |  |          |  +--:(dynamic)
      |        |  |          |  |  +--ro dynamic?                empty
      |        |  |          |  +--:(explicit)
      |        |  |          |     +--ro explicit-path-name? leafref
      |        |  |          +--ro no-cspf?                 empty
      |        |  |          +--ro lockdown?                empty
      |        |  +--:(p2mp) {ietf-te-types:p2mp-te}?
      |        |     +--ro p2mp-paths* [destination]
      |        |        +--ro destination      inet:ip-address
      |        |        +--ro primary-paths* [preference]
      |        |           +--ro preference            uint8
      |        |           +--ro tunnel-path-params
      |        |           |  +--ro path-named-constraint?   leafref
      |        |           |  +--ro path-selection
      |        |           |  |  +--ro topology?        topology-id
      |        |           |  |  +--ro cost-limit?      uint32
      |        |           |  |  +--ro hop-limit?       uint8
      |        |           |  |  +--ro metric-type?     identityref
      |        |           |  |  +--ro tiebreaker-type? identityref
```

```
      |        |           |  |  +--ro ignore-overload? boolean
      |        |           |  |  +--ro tunnel-path-affinities
      |        |           |  |  |  +--ro (style)?
      |        |           |  |  |     +--:(values)
      |        |           |  |  |     |  +--ro value?         uint32
      |        |           |  |  |     |  +--ro mask?          uint32
      |        |           |  |  |     +--:(named)
      |        |           |  |  |        +--ro constraints* [usage]
      |        |           |  |  |           +--ro usage identityref
      |        |           |  |  |           +--ro constraint
      |        |           |  |  |              +--ro affinity-names*
      |        |           |  |  |                 +--ro name    string
      |        |           |  |  +--ro tunnel-path-srlgs
      |        |           |  |     +--ro (style)?
      |        |           |  |        +--:(values)
      |        |           |  |        |  +--ro usage? identityref
      |        |           |  |        |  +--ro values*        srlg
      |        |           |  |        +--:(named)
      |        |           |  |           +--ro constraints* [usage]
      |        |           |  |              +--ro usage identityref
      |        |           |  |              +--ro constraint
      |        |           |  |                 +--ro srlg-names* [name]
      |        |           |  |                    +--ro name    string
      |        |           |  +--ro (type)?
      |        |           |  |  +--:(dynamic)
      |        |           |  |  |  +--ro dynamic?                 empty
      |        |           |  |  +--:(explicit)
      |        |           |  |     +--ro explicit-path-name? leafref
      |        |           |  +--ro no-cspf?                 empty
      |        |           |  +--ro lockdown?                empty
      |        |           +--ro seondary-paths* [preference]
      |        |              +--ro preference            uint8
      |        |              +--ro tunnel-path-params
      |        |                 +--ro path-named-constraint?   leafref
      |        |                 +--ro path-selection
      |        |                 |  +--ro topology?        topology-id
      |        |                 |  +--ro cost-limit?      uint32
      |        |                 |  +--ro hop-limit?       uint8
      |        |                 |  +--ro metric-type?     identityref
      |        |                 |  +--ro tiebreaker-type? identityref
      |        |                 |  +--ro ignore-overload? boolean
      |        |                 |  +--ro tunnel-path-affinities
      |        |                 |  |  +--ro (style)?
      |        |                 |  |     +--:(values)
      |        |                 |  |     |  +--ro value?         uint32
      |        |                 |  |     |  +--ro mask?          uint32
      |        |                 |  |     +--:(named)
      |        |                 |  |        +--ro constraints* [usage]
```

```
      |        |                 |  |           +--ro usage identityref
      |        |                 |  |           +--ro constraint
      |        |                 |  |              +--ro affinity-names*
      |        |                 |  |                 +--ro name string
      |        |                 |  +--ro tunnel-path-srlgs
      |        |                 |     +--ro (style)?
      |        |                 |        +--:(values)
      |        |                 |        |  +--ro usage? identityref
      |        |                 |        |  +--ro values*        srlg
      |        |                 |        +--:(named)
      |        |                 |           +--ro constraints* [usage]
      |        |                 |              +--ro usage identityref
      |        |                 |              +--ro constraint
      |        |                 |                 +--ro srlg-names*
      |        |                 |                    +--ro name string
      |        |                 +--ro (type)?
      |        |                 |  +--:(dynamic)
      |        |                 |  |  +--ro dynamic?            empty
      |        |                 |  +--:(explicit)
      |        |                 |     +--ro explicit-path-name? leafref
      |        |                 +--ro no-cspf?                 empty
      |        |                 +--ro lockdown?                empty
      |        +--ro oper-status?            identityref
      |        +--ro lsp* [source destination tunnel-id lsp-id]
      |           +--ro source                leafref
      |           +--ro destination           leafref
      |           +--ro tunnel-id             leafref
      |           +--ro lsp-id                leafref
      |           +--ro extended-tunnel-id?   leafref
      |           +--ro type?                 leafref
      +--ro tunnels-state
      |  +--ro tunnel* [name type]
      |     +--ro name          string
      |     +--ro type          identityref
      |     +--ro identifier?   uint16
```

Figure 8: TE tunnels configuration and state tree

3.4. TE LSPs State Data

TE LSPs are derived state data that is usually instantiated via signaling protocols. TE LSPs exists on routers as ingress (starting point of LSP), transit (mid-point of LSP), or egress (termination point of the LSP). TE LSPs are distinguished by the 5 tuple, and LSP type (P2P or P2MP) as shown in Figure 9.

In the model, the nodes holding LSPs data exist in a read-only list as shown below:

```
         +--ro lsps-state
            +--ro lsp* [source destination tunnel-id lsp-id
                        extended-tunnel-id type]
               +--ro source                    inet:ip-address
               +--ro destination               inet:ip-address
               +--ro tunnel-id                 uint16
               +--ro lsp-id                    uint16
               +--ro extended-tunnel-id        inet:ip-address
               +--ro type                      identityref
               +--ro oper-status?              identityref
               +--ro origin-type?              enumeration
               +--ro lsp-resource-status?      enumeration
               +--ro lsp-protection-status?    enumeration
               +--ro lsp-operational-status?   empty
               +--ro lsp-timers
               |  +--ro life-time?         uint32
               |  +--ro time-to-install?   uint32
               |  +--ro time-to-die?       uint32
               +--ro downstream-info
               |  +--ro nhop?                 inet:ip-address
               |  +--ro outgoing-interface?   if:interface-ref
               |  +--ro neighbor?             inet:ip-address
               |  +--ro label?                uint32
               +--ro upstream-info
                  +--ro nhop?                 inet:ip-address
                  +--ro incoming-interface?   if:interface-ref
                  +--ro neighbor?             inet:ip-address
                  +--ro label?                uint32
```

Figure 9: TE LSPs state tree

3.5. Global RPC Data

This branch of the model covers system-wide RPC execution data to trigger actions and optionally expect responses. Examples of such TE commands are to:

- o Clear global TE statistics of various features

3.6. Interface RPC Data

This collection of data in the model defines TE interface RPC execution commands. Examples of these are to:

- o Clear TE statistics for all or for individual TE interfaces
- o Trigger immediate flooding for one or all TE interfaces

3.7. Tunnel RPC Data

This branch of the model covers TE tunnel RPC execution data to trigger actions and optionally expect responses. Examples of such TE commands are:

- o Clear statistics for all or for individual tunnels

3.8. Global Notifications Data

This branch of the model covers system-wide notifications data. The node notifies the registered events to the server using the defined notification messages. Example of such global TE events are:

- o Backup tunnel FRR active and not-active state transition events

3.9. Interfaces Notifications Data

This branch of the model covers TE interfaces related notifications data. The TE interface configuration is used for specific events registration. Notifications are sent for registered events to the server. Example events for TE interfaces are:

- o Interface creation and deletion
- o Interface state transitions
- o (Soft) preemption triggers
- o Fast reroute activation

3.10. Tunnel Notification Data

This branch of the model covers TE tunnels related notifications data. The TE tunnels configuration is used for specific events registration. Notifications are sent for registered events to the server. Example events for TE tunnels are:

- o Tunnel creation and deletion events
- o Tunnel state up/down changes
- o Tunnel state reoptimization changes

4. TE Generic and Helper YANG Modules

```
<CODE BEGINS>file "ietf-te-types@2015-07-06.yang"
module ietf-te-types {

  namespace "urn:ietf:params:xml:ns:yang:ietf-te-types";

  /* Replace with IANA when assigned */
  prefix "te-types";

  import ietf-inet-types {
    prefix inet;
  }

  organization
    "IETF TEAS Working Group";

  contact "Fill me";

  description
    "This module contains a collection of generally
    useful TE specific YANG data type defintions.";

  revision 2015-07-06 {
    description "Latest revision of TE basic types";
    reference "RFC3209";
  }

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

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

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

  identity state-type {
    description
```

```
        "Base identity for TE states";
    }

    identity state-up {
      base state-type;
      description
        "State up";
    }

    identity state-down {
      base state-type;
      description
        "State down";
    }

    identity switching-capabilities {
      description
        "Base identity for interface switching capabilities";
    }

    identity switching-psc1 {
      base switching-capabilities;
      description
        "Packet-Switch Capable-1 (PSC-1)";
    }

    identity switching-evpl {
      base switching-capabilities;
      description
        "Ethernet Virtual Private Line (EVPL)";
    }

    identity switching-l2sc {
      base switching-capabilities;
      description
        "Layer-2 Switch Capable (L2SC)";
    }

    identity switching-tdm {
      base switching-capabilities;
      description
        "Time-Division-Multiplex Capable (TDM)";
    }

    identity switching-otn {
      base switching-capabilities;
      description
        "OTN-TDM capable";
```

```
     }

     identity switching-dcsc {
       base switching-capabilities;
       description
         "Data Channel Switching Capable (DCSC)";
     }

     identity switching-lsc {
       base switching-capabilities;
       description
         "Lambda-Switch Capable (LSC)";
     }

     identity switching-fsc {
       base switching-capabilities;
       description
         "Fiber-Switch Capable (FSC)";
     }

     identity lsp-encoding-types {
       description
         "Base identity for encoding types";
     }

     identity lsp-encoding-packet {
       base lsp-encoding-types;
       description
         "Packet LSP encoding";
     }

     identity lsp-encoding-ethernet {
       base lsp-encoding-types;
       description
         "Ethernet LSP encoding";
     }

     identity lsp-encoding-pdh {
       base lsp-encoding-types;
       description
         "ANSI/ETSI LSP encoding";
     }

     identity lsp-encoding-sdh {
       base lsp-encoding-types;
       description
         "SDH ITU-T G.707 / SONET ANSI T1.105 LSP encoding";
     }
```

```
identity lsp-encoding-digital-wrapper {
  base lsp-encoding-types;
  description
    "Digital Wrapper LSP encoding";
}

identity lsp-encoding-lambda {
  base lsp-encoding-types;
  description
    "Lambda (photonic) LSP encoding";
}

identity lsp-encoding-fiber {
  base lsp-encoding-types;
  description
    "Fiber LSP encoding";
}

identity lsp-encoding-fiber-channel {
  base lsp-encoding-types;
  description
    "FiberChannel LSP encoding";
}

identity lsp-encoding-oduk {
  base lsp-encoding-types;
  description
    "G.709 ODUk (Digital Path)LSP encoding";
}

identity lsp-encoding-optical-channel {
  base lsp-encoding-types;
  description
    "Line (e.g., 8B/10B) LSP encoding";
}

identity lsp-encoding-line {
  base lsp-encoding-types;
  description
    "Line (e.g., 8B/10B) LSP encoding";
}

/* TE basic features */
feature p2mp-te {
  description
    "Indicates support for P2MP-TE";
}
```

```
     feature frr-te {
       description
         "Indicates support for TE FastReroute (FRR)";
     }

     feature extended-admin-groups {
       description
         "Indicates support for TE link extended admin
         groups.";
     }

     feature named-path-affinities {
       description
         "Indicates support for named path affinities";
     }

     feature named-extended-admin-groups {
       description
         "Indicates support for named extended admin groups";
     }

     feature named-srlg-groups {
       description
         "Indicates support for named SRLG groups";
     }

     feature named-path-constraints {
       description
         "Indicates support for named path constraints";
     }

     grouping explicit-route-subobject {
       description
         "The explicit route subobject grouping";
       choice type {
         description
           "The explicit route subobject type";
         case ipv4-address {
           description
             "IPv4 address explicit route subobject";
           leaf v4-address {
             type inet:ipv4-address;
             description
               "An IPv4 address.  This address is
               treated as a prefix based on the
               prefix length value below. Bits beyond
               the prefix are ignored on receipt and
               SHOULD be set to zero on transmission.";
```

```
          }
          leaf v4-prefix-length {
            type uint8;
            description
              "Length in bits of the IPv4 prefix";
          }
          leaf v4-loose {
            type boolean;
            description
              "Describes whether the object is loose
              if set, or otherwise strict";
          }
        }
        case ipv6-address {
          description
            "IPv6 address Explicit Route Object";
          leaf v6-address {
            type inet:ipv6-address;
            description
              "An IPv6 address.  This address is
              treated as a prefix based on the
              prefix length value below.  Bits
              beyond the prefix are ignored on
              receipt and SHOULD be set to zero
              on transmission.";
          }
          leaf v6-prefix-length {
            type uint8;
            description
              "Length in bits of the IPv4 prefix";
          }
          leaf v6-loose {
            type boolean;
            description
              "Describes whether the object is loose
              if set, or otherwise strict";
          }
        }
        case as-number {
          leaf as-number {
            type uint16;
            description "AS number";
          }
          description
            "Autonomous System explicit route subobject";
        }
        case unnumbered-link {
          leaf router-id {
```

```
            type inet:ip-address;
            description
              "A router-id address";
          }
          leaf interface-id {
            type uint32;
            description "The interface identifier";
          }
          description
            "Unnumbered link explicit route subobject";
          reference
            "RFC3477: Signalling Unnumbered Links in
            RSVP-TE";
        }
        case label {
          leaf value {
            type uint32;
            description "the label value";
          }
          description
            "The Label ERO subobject";
        }
        /* AS domain sequence..? */
      }
    }

    grouping record-route-subobject {
      description
        "The record route subobject grouping";
      choice type {
        description
          "The record route subobject type";
        case ipv4-address {
          leaf v4-address {
            type inet:ipv4-address;
            description
              "An IPv4 address.  This address is
              treated as a prefix based on the prefix
              length value below. Bits beyond the
              prefix are ignored on receipt and
              SHOULD be set to zero on transmission.";
          }
          leaf v4-prefix-length {
            type uint8;
            description
              "Length in bits of the IPv4 prefix";
          }
          leaf v4-flags {
```

```
            type uint8;
            description
              "IPv4 address sub-object flags";
            reference "RFC3209";
          }
        }
        case ipv6-address {
          leaf v6-address {
            type inet:ipv6-address;
            description
              "An IPv6 address.  This address is
              treated as a prefix based on the
              prefix length value below. Bits
              beyond the prefix are ignored on
              receipt and SHOULD be set to zero
              on transmission.";
          }
          leaf v6-prefix-length {
            type uint8;
            description
              "Length in bits of the IPv4 prefix";
          }
          leaf v6-flags {
            type uint8;
            description
              "IPv6 address sub-object flags";
            reference "RFC3209";
          }
        }
        case label {
          leaf value {
            type uint32;
            description "the label value";
          }
          leaf flags {
            type uint8;
            description
              "Label sub-object flags";
            reference "RFC3209";
          }
          description
            "The Label ERO subobject";
        }
      }
    }

    identity route-usage-type {
      description
```

```
        "Base identity for route usage";
    }

    identity route-include-ero {
      base route-usage-type;
      description
        "Include ERO from route";
    }

    identity route-exclude-ero {
      base route-usage-type;
      description
        "Exclude ERO from route";
    }

    identity route-exclude-srlg {
      base route-usage-type;
      description
        "Exclude SRLG from route";
    }

    identity path-metric-type {
      description
        "Base identity for path metric type";
    }

    identity path-metric-te {
      base path-metric-type;
      description
        "TE path metric";
    }

    identity path-metric-igp {
      base path-metric-type;
      description
        "IGP path metric";
    }

    identity path-tiebreaker-type {
      description
        "Base identity for path tie-breaker type";
    }

    identity path-tiebreaker-minfill {
      base path-tiebreaker-type;
      description
        "Min-Fill LSP path placement";
    }
```

```
     identity path-tiebreaker-maxfill {
       base path-tiebreaker-type;
       description
         "Max-Fill LSP path placement";
     }

     identity path-tiebreaker-randoom {
       base path-tiebreaker-type;
       description
         "Random LSP path placement";
     }

     identity bidir-provisioning-mode {
       description
         "Base identity for bidirectional provisioning
         mode.";
     }

     identity bidir-provisioning-single-sided {
       base bidir-provisioning-mode;
       description
         "Single-sided bidirectional provioning mode";
     }

     identity bidir-provisioning-double-sided {
       base bidir-provisioning-mode;
       description
         "Double-sided bidirectional provioning mode";
     }

     identity bidir-association-type {
       description
         "Base identity for bidirectional association type";
     }

     identity bidir-assoc-corouted {
       base bidir-association-type;
       description
         "Co-routed bidirectional association type";
     }

     identity bidir-assoc-non-corouted {
       base bidir-association-type;
       description
         "Non co-routed bidirectional association type";
     }

     identity resource-affinities-type {
```

```
      description
        "Base identity for resource affinities";
    }

    identity resource-aff-include-all {
      base resource-affinities-type;
      description
        "The set of attribute filters associated with a
        tunnel all of which must be present for a link
        to be acceptable";
    }

    identity resource-aff-include-any {
      base resource-affinities-type;
      description
        "The set of attribute filters associated with a
        tunnel any of which must be present for a link
        to be acceptable";
    }

    identity resource-aff-exclude-any {
      base resource-affinities-type;
      description
        "The set of attribute filters associated with a
        tunnel any of which renders a link unacceptable";
    }

    typedef admin-group {
      type binary {
        length 32;
      }
      description
        "Administrative group/Resource class/Color.";
    }

    typedef extended-admin-group {
      type binary;
      description
        "Extended administrative group/Resource class/Color.";
    }

    typedef admin-groups {
      type union {
        type admin-group;
        type extended-admin-group;
      }
      description "TE administrative group derived type";
    }
```

```
typedef srlg {
  type uint32;
  description "SRLG type";
}

identity path-computation-srlg-type {
  description
    "Base identity for SRLG path computation";
}

identity srlg-ignore {
  base path-computation-srlg-type;
  description
    "Ignores SRLGs in path computation";
}

identity srlg-strict {
  base path-computation-srlg-type;
  description
    "Include strict SRLG check in path computation";
}

identity srlg-preferred {
  base path-computation-srlg-type;
  description
    "Include preferred SRLG check in path computation";
}

identity srlg-weighted {
  base path-computation-srlg-type;
  description
    "Include weighted SRLG check in path computation";
}

typedef te-metric {
  type uint32;
  description
    "TE link metric";
}

typedef topology-id {
  type string {
    pattern '/?([a-zA-Z0-9\-_.]+)(/[a-zA-Z0-9\-_.]+)*';
  }
  description
    "An identifier for a topology.";
}
```

```
     /**
      * TE tunnel generic groupings
      **/

     /* Tunnel path selection parameters */
     grouping tunnel-path-selection {
       description
         "Tunnel path selection properties grouping";
       container path-selection {
         description
           "Tunnel path selection properties container";
         leaf topology {
           type topology-id;
           description
             "The tunnel path is computed using the specific
             topology identified by this identifier";
         }
         leaf cost-limit {
           type uint32 {
             range "1..4294967295";
           }
           description
             "The tunnel path cost limit.";
         }
         leaf hop-limit {
           type uint8 {
             range "1..255";
           }
           description
             "The tunnel path hop limit.";
         }
         leaf metric-type {
           type identityref {
             base path-metric-type;
           }
           default path-metric-te;
           description
             "The tunnel path metric type.";
         }
         leaf tiebreaker-type {
           type identityref {
             base path-tiebreaker-type;
           }
           default path-tiebreaker-maxfill;
           description
             "The tunnel path computation tie breakers.";
         }
         leaf ignore-overload {
```

```
          type boolean;
          description
            "The tunnel path can traverse overloaded node.";
        }
        uses tunnel-path-affinities;
        uses tunnel-path-srlgs;
      }
    }

    grouping tunnel-path-affinities {
      description
        "Path affinities grouping";
      container tunnel-path-affinities {
        if-feature named-path-affinities;
        description
          "Path affinities container";
        choice style {
          description
            "Path affinities representation style";
          case values {
            leaf value {
              type uint32 {
                range "0..4294967295";
              }
              description
                "Affinity value";
            }
            leaf mask {
              type uint32 {
                range "0..4294967295";
              }
              description
                "Affinity mask";
            }
          }
          case named {
            list constraints {
              key "usage";
              leaf usage {
                type identityref {
                  base resource-affinities-type;
                }
                description "Affinities usage";
              }
              container constraint {
                description
                  "Container for named affinities";
                list affinity-names {
```

```
                    key "name";
                    leaf name {
                      type string;
                      description
                        "Affinity name";
                    }
                    description
                      "List of named affinities";
                  }
                }
                description
                  "List of named affinity constraints";
              }
            }
          }
        }
      }

      grouping tunnel-path-srlgs {
        description
          "Path SRLG properties grouping";
        container tunnel-path-srlgs {
          description
            "Path SRLG properties container";
          choice style {
            description
              "Type of SRLG representation";
            case values {
              leaf usage {
                type identityref {
                  base route-exclude-srlg;
                }
                description "SRLG usage";
              }
              leaf-list values {
                type srlg;
                description "SRLG value";
              }
            }
            case named {
              list constraints {
                key "usage";
                leaf usage {
                  type identityref {
                    base route-exclude-srlg;
                  }
                  description "SRLG usage";
                }
```

```
               container constraint {
                 description
                   "Container for named SRLG list";
                 list srlg-names {
                   key "name";
                   leaf name {
                     type string;
                     description
                       "The SRLG name";
                   }
                   description
                     "List named SRLGs";
                 }
               }
               description
                 "List of named SRLG constraints";
             }
           }
         }
       }
     }

     grouping tunnel-bidir-assoc-properties {
       description
         "TE tunnel associated bidirectional properties
         grouping";
       container bidirectional {
         description
           "TE tunnel associated bidirectional attributes.";
         container association {
           description
             "Tunnel bidirectional association properties";
           leaf id {
             type uint16;
             description
               "The TE tunnel association identifier.";
           }
           leaf source {
             type inet:ip-address;
             description
               "The TE tunnel association source.";
           }
           leaf global-source {
             type inet:ip-address;
             description
               "The TE tunnel association global
               source.";
           }
```

```
          leaf type {
            type identityref {
              base bidir-association-type;
            }
            default bidir-assoc-non-corouted;
            description
              "The TE tunnel association type.";
          }
          leaf provisioing {
            type identityref {
              base bidir-provisioning-mode;
            }
            description
              "Describes the provisioning model of the
              associated bidirectional LSP";
            reference
              "draft-ietf-teas-mpls-tp-rsvpte-ext-
              associated-lsp, section-3.2";
          }
        }
      }
    }
    /*** End of TE tunnel groupings ***/

    /**
     * TE interface generic groupings
     **/

    /* TE interface flooding parameters */
    grouping interface-te-flooding-parameters_config {
      description "Interface TE flooding properties.";
      container thresholds {
        description "Flooding threshold values in percentages.";
        choice type {
          description
            "Describes the flooding threshold step method";
          case equal-steps {
            choice equal-step-type {
              description
                "Describes whether up and down equal step
                size are same or different";
              case up-down-different-step {
                leaf up-step {
                  type uint8 {
                    range "0..100";
                  }
                  description
                    "Set single percentage threshold
```

```
                    for increasing resource
                      allocation";
                }
                leaf down-step {
                  type uint8 {
                    range "0..100";
                  }
                  description
                    "Set single percentage threshold
                    for decreasing resource
                      allocation";
                }
              }
              case up-down-same-step {
                leaf step {
                  type uint8 {
                    range "0..100";
                  }
                  description
                    "Set single percentage threshold
                    for increasing and decreasing
                      resource allocation";
                }
              }
            }
          }
          case unequal-steps {
            list up-steps {
              key "value";
              description
                "Set nultuple percentage thresholds for
                increasing resource allocation";
              leaf value {
                type uint8 {
                  range "0..100";
                }
                description
                  "Percentage value";
              }
            }
            list down-steps {
              key "value";
              description
                "Set nultuple percentage thresholds for
                decreasing resource allocation";
              leaf value {
                type uint8 {
                  range "0..100";
```

```
                  }
                  description
                    "Percentage value";
                }
              }
            }
          }
        }
      }

      grouping interface-te-flooding-parameters {
        description "Interface TE flooding properties.";
        container te-flooding-parameters {
          description
            "Parameters for interface flooding";
          container config {
            description
              "Configuration parameters for interface
              flooding";
            uses interface-te-flooding-parameters_config;
          }
          container state {
            config false;
            description
              "State parameters for interface flooding";
            uses interface-te-flooding-parameters_config;
          }
        }
      }
    }
    <CODE ENDS>
```

Figure 10: TE basic types YANG module

```
    <CODE BEGINS> file "ietf-te-psc-types@2015-07-06.yang"
    module ietf-te-psc-types {

      namespace "urn:ietf:params:xml:ns:yang:ietf-te-psc-types";

      /* Replace with IANA when assigned */
      prefix "te-psc-types";

      import ietf-inet-types { prefix inet; }

      organization
        "IETF TEAS Working Group";

      contact "Fill me";
```

```
     description
       "This module contains a collection of generally
       useful TE specific YANG data type defintions.";

     revision 2015-07-06 {
       description "Latest revision of TE MPLS/packet types";
       reference "RFC3209";
     }

     /* Describes egress LSP label allocation */
     typedef egress-label {
       type enumeration {
         enum "IPv4-EXPLICIT-NULL" {
           description
             "Use IPv4 explicit-NULL MPLS label at the
             egress";
         }
         enum "IPv6-EXPLICIT-NULL" {
           description
             "Use IPv6 explicit-NULL MPLS label at the
             egress";
         }
         enum "IMPLICIT-NULL" {
           description
             "Use implicit-NULL MPLS label at the egress";
         }
         enum "NON-NULL"{
           description
           "Use a non NULL MPLS label at the egress";
       }
       }
       description
         "Describes egress label allocation";
     }

     identity backup-type {
       description
         "Base identity for backup protection types";
     }

     identity backup-facility {
       base backup-type;
       description
         "Use facility backup to protect LSPs traversing
         protected TE interface";
       reference
         "RFC49090: RSVP-TE Fast Reroute";
     }
```

```
identity backup-detour {
  base backup-type;
  description
    "Use detour or 1-for-1 protection";
  reference
    "RFC49090: RSVP-TE Fast Reroute";
}

identity backup-protection-type {
  description
    "Base identity for backup protection type";
}

identity backup-protection-link {
  base backup-protection-type;
  description
    "backup provides link protection only";
}

identity backup-protection-node-link {
  base backup-protection-type;
  description
    "backup offers node (preferred) or link protection";
}

identity bc-model-type {
  description
    "Base identity for Diffserv-TE bandwidth constraint
    model type";
}

identity bc-model-rdm {
  base bc-model-type;
  description
    "Russian Doll bandwidth constraint model type.";
}

identity bc-model-mam {
  base bc-model-type;
  description
    "Maximum Allocation bandwidth constraint
    model type.";
}

identity bc-model-mar {
  base bc-model-type;
  description
    "Maximum Allocation with Reservation
```

```
          bandwidth constraint model type.";
      }

      grouping bandwidth-constraint-values {
        description
          "Packet bandwidth contraints values";
        choice value-type {
          description
            "Value representation";
          case percentages {
            container perc-values {
              uses bandwidth-psc-constraints;
              description
                "Percentage values";
            }
          }
          case absolutes {
            container abs-values {
              uses bandwidth-psc-constraints;
              description
                "Absolute values";
            }
          }
        }
      }

      grouping bandwidth-psc-reservable {
        description
          "Packet reservable bandwidth";
        choice bandwidth-value {
          description "Reservable bandwidth configuraiton choice";
          case absolute {
            leaf absolute-value {
              type uint32;
              description "Absolute value of the bandwidth";
            }
          }
          case precentage {
            leaf percent-value {
              type uint32 {
                range "0..4294967295";
              }
              description "Percentage reservable bandwidth";
            }
            description
              "The maximum reservable bandwidth on the
              interface";
          }
```

```
        }
        choice bc-model-type {
          description
            "Reservable bandwidth percentage capacity
            values.";
          case bc-model-rdm {
            container bc-model-rdm {
              description
                "Russian Doll Model Bandwidth Constraints.";
              uses bandwidth-psc-constraints;
            }
          }
          case bc-model-mam {
            container bc-model-mam {
              uses bandwidth-psc-constraints;
              description
                "Maximum Allocation Model Bandwidth
                Constraints.";
            }
          }
          case bc-model-mar {
            container bc-model-mar {
              uses bandwidth-psc-constraints;
              description
                "Maximum Allocation with Reservation Model
                Bandwidth Constraints.";
            }
          }
        }
      }

      typedef bfd-type {
        type enumeration {
          enum classical {
            description "BFD classical session type.";
          }
          enum seamless {
            description "BFD seamless session type.";
          }
        }
        default "classical";
        description
          "Type of BFD session";
      }

      typedef bfd-encap-mode-type {
        type enumeration {
          enum gal {
```

```
            description
              "BFD with GAL mode";
          }
          enum ip {
            description
              "BFD with IP mode";
          }
        }
        default ip;
        description
          "Possible BFD transport modes when running over TE
          LSPs.";
      }

      grouping bandwidth-psc-constraints {
        description "Bandwidth constraints.";
        container bandwidth-psc-constraints {
          description
            "Holds the bandwidth contraints properties";
          leaf maximum-reservable {
            type uint32 {
              range "0..4294967295";
            }
            description
              "The maximum reservable bandwidth on the
              interface";
          }
          leaf-list bc-value {
            type uint32 {
              range "0..4294967295";
            }
            max-elements 8;
            description
              "The bandwidth contraint type";
          }
        }
      }

      grouping tunnel-forwarding-properties {
        description "Properties for using tunnel in forwarding.";
        container forwarding {
          description
            "Tunnel forwarding properties container";
          leaf load-share {
            type uint32 {
              range "1..4294967295";
            }
            description "ECMP tunnel forwarding
```

```
            load-share factor.";
        }
        choice policy-type {
          description
            "Tunnel policy type";
          container class {
            description
              "Tunnel forwarding per class properties";
            leaf class {
              type uint8 {
                range "1..7";
              }
              description
                "The class associated with this tunnel";
            }
          }
          container group {
            description
              "Tunnel frowarding per group properties";
            leaf-list classes {
              type uint8 {
                range "1..7";
              }
              description
                "The forwarding class";
            }
          }
        }
      }
    }

    grouping tunnel-routing-properties {
      description
        "TE tunnel routing properties";
      choice routing-choice {
        description
          "Announces the tunnel to IGP as either
          autoroute or forwarding adjacency.";
        case autoroute {
          container autoroute-announce {
            presence "Enable autoroute announce.";
            description
              "Announce the TE tunnel as autoroute to
              IGP for use as IGP shortcut.";
            leaf-list routing-afs {
              type inet:ip-version;
              description
                "Address families";
```

```
              }
              choice metric-type {
                description
                  "Type of metric to use when announcing
                  the tunnel as shortcut";
                leaf metric {
                  type uint32 {
                    range "1..2147483647";
                  }
                  description
                    "Describes the metric to use when
                    announcing the tunnel as shortcut";
                }
                leaf relative-metric {
                  type int32 {
                    range "-10..10";
                  }
                  description
                    "Relative TE metric to use when
                    announcing the tunnel as shortcut";
                }
                leaf absolute-metric {
                  type uint32 {
                    range "1..2147483647";
                  }
                  description
                    "Absolute TE metric to use when
                    announcing the tunnel as shortcut";
                }
              }
            }
          }
          case forwarding-adjacency {
            container forwarding-adjacency {
              presence "Enable forwarding adjacency
                on the tunnel.";
              description
                "Announce the TE tunnel
                as forwarding adjacency.";
              leaf holdtime {
                type uint32 {
                  range "0..4294967295";
                }
                description
                  "Holdtime in seconds after
                  tunnel becomes UP.";
              }
              leaf-list routing-afs {
```

```
               type inet:ip-version;
               description
                 "Address families";
             }
           }
         }
       }
     }
   }
   <CODE ENDS>
```

Figure 11: TE packet/MPLS specific types YANG module

```
   <CODE BEGINS> file "ietf-te@2015-07-06.yang"
   module ietf-te {

     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 ietf-te-types;
     }

     /* Import TE packet specific types */
     import ietf-te-psc-types {
       prefix ietf-te-psc-types;
     }

     import ietf-interfaces {
       prefix if;
     }

     import ietf-inet-types {
       prefix inet;
     }

     organization
       "IETF TEAS Working Group";

     contact
       "Fill me";

     description
       "YANG data module for TE configuration,
       state, RPC and notifications.";
```

```
     revision 2015-07-06 {
       description "Latest update to TE generic YANG module.";
       reference "TBD";
     }

     /**
      * TE interface generic groupings
      */
     grouping te-admin-groups_config {
       description
         "TE interface affinities grouping";
       choice admin-group-type {
         description
           "TE interface administrative groups
           representation type";
         case value-admin-groups {
           choice value-admin-group-type {
             description "choice of admin-groups";
             case value-admin-groups {
               description
                 "Administrative group/Resource
                 class/Color.";
               leaf admin-group {
                 type ietf-te-types:admin-group;
                 description
                   "TE interface administrative group";
               }
             }
             case value-extended-admin-groups {
               if-feature ietf-te-types:extended-admin-groups;
               description
                 "Extended administrative group/Resource
                 class/Color.";
               leaf extended-admin-group {
                 type ietf-te-types:extended-admin-group;
                 description
                   "TE interface extended administrativei
                   group";
               }
             }
           }
         }
         case named-admin-groups {
           list named-admin-groups {
             if-feature ietf-te-types:extended-admin-groups;
             if-feature ietf-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/" +
                  "named-admin-groups/config/"+
                  "named-admin-groups/name";
              }
              description
                "A named admin-group entry";
            }
          }
        }
      }
    }

    grouping te-admin-groups {
      description "TE admin-group configuration grouping";
      container te-admin-groups {
        description
          "Configuration parameters for interface
          administrative groups";
        container config {
          description
            "Configuration parameters for interface
            administrative groups";
          uses te-admin-groups_config;
        }
        container state {
          config false;
          description
            "Configuration parameters for interface
            administrative groups";
          uses te-admin-groups_config;
        }
      }
    }

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

      grouping te-srlgs {
        description "TE SRLG configuration grouping";
        container te-srlgs {
          description "TBD";
          container config {
            description
              "Configuration parameters for interface SRLG
              groups";
            uses te-srlgs_config;
          }
          container state {
            config false;
            description
              "State parameters for interface SRLG groups";
            uses te-srlgs_config;
          }
        }
      }

      grouping te-metric_config {
        description "Interface TE metric grouping";
```

```
        leaf te-metric {
          type ietf-te-types:te-metric;
          description "Interface TE metric.";
        }
      }

      grouping te-attributes {
        description "TE attributes configuration grouping";
        container config {
          description
            "Configuration parameters for interface TE
             attributes";
          uses te-metric_config;
        }
        container state {
          config false;
          description
            "State parameters for interface TE metric";
          uses te-metric_config;
          uses interface-advertisements_state;
        }
      }

      /* 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 ietf-te-types:switching-capabilities;
            }
            description
              "Switching Capability for this interface";
          }
          leaf encoding {
            type identityref {
              base ietf-te-types:lsp-encoding-types;
            }
            description
              "Encoding supported by this interface";
          }
        }
      }
```

```
     grouping te-switching-cap {
       description "TE interface switching capability grouping";
       container te-switching-cap {
         description
           "Interface switching capabilities container";
         container config {
           description
             "Configuration parameters for interface
             switching capabilities";
           uses te-switching-cap_config;
         }
         container state {
           config false;
           description
             "State parameters for interface switching
             capabilities";
           uses te-switching-cap_config;
         }
       }
     }

     grouping interface-advertisements_state {
       description
         "TE interface advertisements state grouping";
       container interface-advertisements_state {
         description
           "TE interface advertisements state container";
         leaf flood-interval {
           type uint32;
           description
             "The periodic flooding interval";
         }
         leaf last-flooded-time {
           type uint32;
           units seconds;
           description
             "Time elapsed since last flooding in seconds";
         }
         leaf next-flooded-time {
           type uint32;
           units seconds;
           description
             "Time remained for next flooding in seconds";
         }
         leaf last-flooded-trigger {
           type enumeration {
             enum link-up {
               description "Link-up flooding trigger";
```

```
            }
            enum link-down {
              description "Link-up flooding trigger";
            }
            enum threshold-up {
              description
                "Bandwidth reservation up threshold";
            }
            enum threshold-down {
              description
                "Bandwidth reservation down threshold";
            }
            enum bandwidth-change {
              description "Banwidth capacity change";
            }
            enum user-initiated {
              description "Initiated by user";
            }
            enum srlg-change {
              description "SRLG property change";
            }
            enum periodic-timer {
              description "Periodic timer expired";
            }
          }
          description "Trigger for the last flood";
        }
        list advertized-level-areas {
          key level-area;
          description
            "List of areas the TE interface is advertised
            in";
          leaf level-area {
            type uint32;
            description
              "The IGP area or level where the TE
              interface state is advertised in";
          }
        }
      }
    }
    /*** End of TE interface groupings ***/

    /**
     * TE tunnel generic groupings
     */

    /* TE tunnel path properties */
```

```
     grouping tunnel-path-params {
       description
         "Tunnel path properties grouping";
       container tunnel-path-params {
         description
           "Defines a TE tunnel path properties";
         leaf path-named-constraint {
           if-feature ietf-te-types:named-path-constraints;
           type leafref {
             path "/te/globals/named-path-constraints/config/"+
               "named-path-constraints/name";
           }
           description
             "Reference to a globally defined named path
             constraint set";
         }
         uses ietf-te-types:tunnel-path-selection;
         choice type {
           description
             "Describes the path type";
           case dynamic {
             leaf dynamic {
               type empty;
               description
                 "A CSPF dynamically computed path";
             }
           }
           case explicit {
             leaf explicit-path-name {
               type leafref {
                 path "/te/globals/named-explicit-paths/config/"+
                   "named-explicit-paths/name";
               }
               description
                 "Reference to a globally defined
                 explicit-path";
             }
           }
         }
         leaf no-cspf {
           type empty;
           description
             "Indicates no CSPF is to be attempted on this
             path.";
         }
         leaf lockdown {
           type empty;
           description
```

```
              "Indicates no reoptimization to be attempted for
              this path.";
          }
        }
      }

      grouping tunnel-properties_config {
        description
          "Configuration parameters relating to TE tunnel";
        leaf description {
          type string;
          description
            "TE tunnel description.";
        }
        leaf admin-status {
          type identityref {
            base ietf-te-types:state-type;
          }
          default ietf-te-types:state-up;
          description "TE tunnel administrative state.";
        }
        uses ietf-te-psc-types:tunnel-routing-properties;
        uses ietf-te-psc-types:tunnel-forwarding-properties;
        uses ietf-te-types:tunnel-bidir-assoc-properties;
        choice path-type {
          description
            "Describes the path type";
          case p2p {
            leaf destination {
              type inet:ip-address;
              description
                "P2P tunnel destination address";
            }
            /* P2P list of path(s) */
            list primary-paths {
              key "preference";
              description
                "List of primary paths for this
                tunnel.";
              leaf preference {
                type uint8 {
                  range "1..255";
                }
                description
                  "Specifies a preference for
                  this path. The lower the
                  number higher the
                  preference";
```

```
            }
            uses tunnel-path-params;
            list seondary-paths {
              key "preference";
              description
                "List of secondary paths for this
                tunnel.";
              leaf preference {
                type uint8 {
                  range "1..255";
                }
                description
                  "Specifies a preference for
                  this path. The lower the
                  number higher the
                  preference";
              }
              uses tunnel-path-params;
            }
          }
        }
        case p2mp {
          if-feature ietf-te-types:p2mp-te;
          list p2mp-paths {
            key "destination";
            description
              "List of destinations and their
              paths.";
            leaf destination {
              type inet:ip-address;
              description
                "P2MP destination leaf address";
            }
            list primary-paths {
              key "preference";
              description
                "List of primary paths";
              leaf preference {
                type uint8 {
                  range "1..255";
                }
                description
                  "Specifies a preference for
                  this path. The lower the
                  number higher the
                  preference";
              }
              uses tunnel-path-params;
```

```
               list seondary-paths {
                 key "preference";
                 description
                   "List of secondary paths";
                 leaf preference {
                   type uint8 {
                     range "1..255";
                   }
                   description
                     "Specifies a preference
                     for this path. The lower
                       the number higher
                         the preference";
                 }
                 uses tunnel-path-params;
               }
             }
           }
         }
       }
     }

     grouping tunnel-properties {
       description
         "Top level grouping for tunnel properties.";
       container config {
         description
           "Configuration parameters relating to
           tunnel properties";
         uses tunnel-properties_config;
       }
       container state {
         config false;
         description
           "State information associated with tunnel
           properties";
         uses tunnel-properties_config;
         uses tunnel-properties_state;
       }
     }

     grouping tunnel-properties_state {
       description
         "State parameters relating to TE tunnel";
       leaf oper-status {
         type identityref {
           base ietf-te-types:state-type;
         }
```

```
        description "TE tunnel operational state.";
      }
      list lsp {
        key "source destination tunnel-id lsp-id";
        description "List of LSPs associated with the tunnel.";

        leaf source {
          type leafref {
            path "/te/lsps-state/lsp/source";
          }
          description
            "Tunnel sender address extracted from
            SENDER_TEMPLATE  object";
          reference "RFC3209";
        }
        leaf destination {
          type leafref {
            path "/te/lsps-state/lsp/destination";
          }
          description
            "Tunnel endpoint address extracted from
            SESSION object";
          reference "RFC3209";
        }
        leaf tunnel-id {
          type leafref {
            path "/te/lsps-state/lsp/tunnel-id";
          }
          description
            "Tunnel identifier used in the SESSION
            that remains constant over the life
            of the tunnel.";
          reference "RFC3209";
        }
        leaf lsp-id {
          type leafref {
            path "/te/lsps-state/lsp/lsp-id";
          }
          description
            "Identifier used in the SENDER_TEMPLATE
            and the FILTER_SPEC that can be changed
            to allow a sender to share resources with
            itself.";
          reference "RFC3209";
        }
        leaf extended-tunnel-id {
          type leafref {
            path "/te/lsps-state/lsp/extended-tunnel-id";
```

```
          }
          description
            "Extended Tunnel ID of the LSP.";
          reference "RFC3209";
        }
        leaf type {
          type leafref {
            path "/te/lsps-state/lsp/type";
          }
          description "LSP type P2P or P2MP";
        }
      }
    }
    /*** End of TE tunnel groupings ***/

    /**
     * LSP related generic groupings
     */

    grouping lsp-properties_state {
      description
        "State parameters relating to LSP";
      leaf oper-status {
        type identityref {
          base ietf-te-types:state-type;
        }
        description "LSP operational state.";
      }

      leaf origin-type {
        type enumeration {
          enum ingress {
            description
              "Origin ingress";
          }
          enum egress {
            description
              "Origin egress";
          }
          enum transit {
            description
              "transit";
          }
        }
        description
          "Origin type of LSP relative to the location
          of the local switch in the path.";
      }
```

```
        leaf lsp-resource-status {
          type enumeration {
            enum primary {
              description
                "A primary LSP is a fully established LSP for
                 which the resource allocation has been committed
                 at the data plane";
            }
            enum secondary {
              description
                "A secondary LSP is an LSP that has been provisioned
                 in the control plane only; e.g. resource allocation
                 has not been committed at the data plane";
            }
          }
          description "LSP resource allocation type";
          reference "rfc4872, section 4.2.1";
        }

        leaf lsp-protection-status {
          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";
            }
          }
          description "LSP role type";
          reference "rfc4872, section 4.2.1";
        }

        leaf lsp-operational-status {
          type empty;
          description
            "This bit is set when a protecting LSP is carrying the normal
             traffic after protection switching";
        }

        container lsp-timers {
          when "../origin-type = 'ingress'" {
```

```
          description "Applicable to ingress LSPs only";
        }
        description "Ingress LSP timers";
        leaf life-time {
          type uint32;
          units seconds;
          description
            "lsp life time";
        }

        leaf time-to-install {
          type uint32;
          units seconds;
          description
            "lsp installation delay time";
        }

        leaf time-to-die {
          type uint32;
          units seconds;
          description
            "lsp expire delay time";
        }
      }

      container downstream-info {
        description
          "downstream information";

        leaf nhop {
          type inet:ip-address;
          description
            "downstream nexthop.";
        }

        leaf outgoing-interface {
          type if:interface-ref;
          description
            "downstream interface.";
        }

        leaf neighbor {
          type inet:ip-address;
          description
            "downstream neighbor.";
        }

        leaf label {
```

```
          type uint32;
          description
            "downstream label.";
        }
      }

      container upstream-info {
        description
          "upstream information";

        leaf nhop { // phop?
          type inet:ip-address;
          description
            "upstream nexthop.";
        }

        leaf incoming-interface {
          type if:interface-ref;
          description
            "upstream interface.";
        }

        leaf neighbor {
          type inet:ip-address;
          description
            "upstream neighbor.";
        }

        leaf label {
          type uint32;
          description
            "upstream label.";
        }
      }
    }
    /*** End of TE LSP groupings ***/

    /**
     * TE global generic groupings
     */

    grouping global-attributes_config {
      description
        "Top level grouping for global config data.";
    }

    grouping global-attributes_state {
      description
```

```
        "Top level grouping for global state data.";
      leaf tunnels-counter {
        type uint32;
        description "Tunnels count";
      }
      leaf lsps-counter {
        type uint32;
        description "Tunnels count";
      }
    }

    grouping global-attributes {
      description "TE Global attributes grouping";
      container config {
        description
          "Global configuration parameters";
        uses global-attributes_config;
      }
      container state {
        config false;
        description
          "Global configuration parameters";
        uses global-attributes_config;
        uses global-attributes_state;
      }
    }

    grouping named-admin-groups_config {
      description
        "Global named administrative groups configuration
        grouping";
      list named-admin-groups {
        if-feature ietf-te-types:extended-admin-groups;
        if-feature ietf-te-types:named-extended-admin-groups;
        key "name";
        description
          "List of named TE admin-groups";
        leaf name {
          type string;
          description
            "A string name that uniquely identifies a TE
            interface named admin-group";
        }
        leaf group {
          type ietf-te-types:admin-groups;
          description
            "An SRLG value";
        }
```

```
      }
    }

    grouping named-admin-groups {
      description
        "Named admin groups grouping";
      container named-admin-groups {
        description
          "Named admin groups container";
        container config {
          description
            "Configuration parameters for named admin
            groups";
          uses named-admin-groups_config;
        }
        container state {
          config false;
          description
            "State parameters for named admin groups";
          uses named-admin-groups_config;
        }
      }
    }

    grouping named-srlgs_config {
      description
        "Global named SRLGs configuration
        grouping";
      list named-srlgs {
        if-feature ietf-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 group {
          type ietf-te-types:srlg;
          description "An SRLG value";
        }
      }
    }

    grouping named-srlgs {
      description
```

```
        "Global named SRLGs grouping";
      container named-srlgs {
        description
          "Named SRLGs container";
        container config {
          description
            "Configuration parameters for named SRLG groups";
          uses named-srlgs_config;
        }
        container state {
          config false;
          description
            "State parameters for named SRLG groups";
          uses named-srlgs_config;
        }
      }
    }

    grouping named-explicit-paths_config {
      description
        "Global explicit path configuration
        grouping";
      list named-explicit-paths {
        key "name";
        description
          "A list of explicit paths";
        leaf name {
          type string;
          description
            "A string name that uniquely identifies an
            explicit path";
        }
        list explicit-route-objects {
          key "index";
          description
            "List of explicit route objects";
          leaf index {
            type uint8 {
              range "0..255";
            }
            description
              "Index of this explicit route object";
          }
          uses ietf-te-types:explicit-route-subobject;
          leaf explicit-route-usage {
            type identityref {
              base ietf-te-types:route-usage-type;
            }
```

```
            description
              "An IP hop action.";
          }
        }
      }
    }

    grouping named-explicit-paths {
      description
        "Global named explicit path grouping";
      container named-explicit-paths {
        description
          "Nmaed explicit paths container";
        container config {
          description
            "Configuration parameters for named explicit
            paths";
          uses named-explicit-paths_config;
        }
        container state {
          config false;
          description
            "State parameters for named explicit paths";
          uses named-explicit-paths_config;
        }
      }
    }

    grouping named-path-constraints_config {
      description
        "Global named path constraints configuration
        grouping";
      list named-path-constraints {
        if-feature ietf-te-types:named-path-constraints;
        key "name";
        description
          "A list of named path constraints";
        leaf name {
          type string;
          description
            "A string name that uniquely identifies a
            path constraint set";
        }
        uses ietf-te-types:tunnel-path-selection;
      }
    }

    grouping named-path-constraints {
```

```
      description
        "Global named path constraints grouping";
      container named-path-constraints {
        description
          "Nmaed explicit paths container";
        container config {
          description
            "Configuration parameters for named explicit
            paths";
          uses named-path-constraints_config;
        }
        container state {
          config false;
          description
            "State parameters for named explicit paths";
          uses named-path-constraints_config;
        }
      }
    }
    /*** End of TE global groupings ***/

    /**
     * TE configurations container
     */
    container te {
      presence "Enable TE feature.";
      description
        "TE global container.";

      /* TE Global Configuration Data */
      container globals {
        description
          "Configuration data for Global System-wide
          Traffic Engineering.";
        uses global-attributes;
        uses named-admin-groups;
        uses named-srlgs;
        uses named-explicit-paths;
        uses named-path-constraints;
      }

      /* TE Interface Configuration Data */
      container interfaces {
        description
          "Configuration data model for TE interfaces.";
        list interface {
          key "interface";
          description "TE interfaces.";
```

```
          leaf interface {
            type if:interface-ref;
            description
              "TE interface name.";
          }
          /* TE interface parameters */
          uses te-attributes;
          uses te-admin-groups;
          uses te-srlgs;
          uses te-switching-cap;
          /* TE interface flooding parameters */
          uses ietf-te-types:interface-te-flooding-parameters;
        }
      }

      /* TE Tunnel Configuration Data */
      container tunnels {
        description
          "Configuration, operational, notification and RPC
          data model for TE tunnels.";

        list tunnel {
          key "name type";
          unique "identifier";
          description "TE tunnel.";
          leaf name {
            type string;
            description "TE tunnel name.";
          }
          leaf type {
            type identityref {
              base ietf-te-types:tunnel-type;
            }
            description "TE tunnel type.";
          }
          leaf identifier {
            type uint16;
            description
              "TE tunnel Identifier.";
          }
          uses tunnel-properties;
        }
      }

      /* TE Tunnel Ephemeral State Data (TBD) */
      container tunnels-state {
        config "false";
        description
```

```
          "Derived state corresponding to ephemeral tunnels";

        list tunnel {
          key "name type";
          unique "identifier";
          description "TE tunnel.";
          leaf name {
            type string;
            description "TE tunnel name.";
          }
          leaf type {
            type identityref {
              base ietf-te-types:tunnel-type;
            }
            description "TE tunnel type.";
          }
          leaf identifier {
            type uint16;
            description
              "TE tunnel Identifier.";
          }

        }
      }

      /* TE LSPs State Data */
      container lsps-state {
        config "false";
        description "MPLS-TE LSP operational state data.";

        list lsp {
          key
            "source destination tunnel-id lsp-id "+
            "extended-tunnel-id type";
          description
            "List of LSPs associated with the tunnel.";
          leaf source {
            type inet:ip-address;
            description
              "Tunnel sender address extracted from
              SENDER_TEMPLATE  object";
            reference "RFC3209";
          }
          leaf destination {
            type inet:ip-address;
            description
              "Tunnel endpoint address extracted from
              SESSION object";
```

```
            reference "RFC3209";
          }
          leaf tunnel-id {
            type uint16;
            description
              "Tunnel identifier used in the SESSION
              that remains constant over the life
              of the tunnel.";
            reference "RFC3209";
          }
          leaf lsp-id {
            type uint16;
            description
              "Identifier used in the SENDER_TEMPLATE
              and the FILTER_SPEC that can be changed
              to allow a sender to share resources with
              itself.";
            reference "RFC3209";
          }
          leaf extended-tunnel-id {
            type inet:ip-address;
            description
              "Extended Tunnel ID of the LSP.";
            reference "RFC3209";
          }
          leaf type {
            type identityref {
              base ietf-te-types:tunnel-type;
            }
            description "The LSP type P2P or P2MP";
          }
          uses lsp-properties_state;
        }
      }
    }

    /* TE Global RPCs/execution Data */
    rpc globals-rpc {
      description
        "Execution data for TE global.";
    }

    /* TE interfaces RPCs/execution Data */
    rpc interfaces-rpc {
      description
        "Execution data for TE interfaces.";
    }
```

```
     /* TE Tunnel RPCs/execution Data */
     rpc tunnels-rpc {
       description
         "TE tunnels RPC nodes";
     }


     /* TE Global Notification Data */
     notification globals-notif {
       description
         "Notification messages for Global TE.";
     }

     /* TE Interfaces Notification Data */
     notification interfaces-notif {
       description
         "Notification messages for TE interfaces.";
     }

     /* TE Tunnel Notification Data */
     notification tunnels-notif {
       description
         "Notification messages for TE tunnels.";
     }
   }
   <CODE ENDS>
```

Figure 12: TE generic YANG module

5. 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 XML: N/A, the requested URI is an XML namespace.

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

URI: urn:ietf:params:xml:ns:yang:ietf-te-psc-types XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-te namespace: urn:ietf:params:xml:ns:yang:ietf-te prefix: ietf-te reference: RFC3209

name: ietf-te-types namespace: urn:ietf:params:xml:ns:yang:ietf-te-types prefix: ietf-te-types reference: RFC3209

name: ietf-te-psc-types namespace: urn:ietf:params:xml:ns:yang:ietf-te-psc-types prefix: ietf-te-psc-types reference: RFC3209

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [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. Following are the subtrees and data nodes and their sensitivity/vulnerability:

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

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

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

7. Acknowledgement

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

8. References

8.1. 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-19 (work in progress), May 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

8.2. Informative References

[I-D.openconfig-mpls-consolidated-model]
George, J., Fang, L., eric.osborne@level3.com, e., and R. Shakir, "MPLS / TE Model for Service Provider Networks", draft-openconfig-mpls-consolidated-model-00 (work in progress), March 2015.

[I-D.openconfig-netmod-opstate]
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-00 (work in progress), March 2015.

[RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC4328] Papadimitriou, D., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Extensions for G.709 Optical Transport Networks Control", RFC 4328, January 2006.

Authors' Addresses

Tarek Saad (editor)
Cisco Systems Inc

Email: tsaad@cisco.com

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Xufeng Liu
Ericsson

Email: xufeng.liu@ericsson.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

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

Network Working Group
Internet Draft
Intended status: Informational
Expires: January2016

Yunbin Xu
CATR

Guoying Zhang
CATR

July 6, 2015

Requirements of Abstract Alarm Report in ACTN architecture
draft-xu-teas-actn-abstract-alarm-report-00.txt

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on January 6, 2016.

Copyright Notice

Abstract

This draft provides a set of requirements for alarm abstraction and report of transport networks in the ACTN architecture.

Table of Contents

1. Introduction

This draft provides a set of requirements for alarm abstraction and report in ACTN architecture. [ACTN-frame] defines the base reference architecture and terminology.

Section 2 provides the requirements for alarm report in ACTN architecture. Section 3 provides some solutions for alarm abstraction of transport networks.

2. Requirements for Alarm Report in ACTN Architecture

In ACTN architecture, Physical Network controller (PNC) can access to all of the network resources and alarm information, and provides an abstracted view of transport network resources to upper layer controller based on different abstraction policy. Multi-Domain Service Controller (MDSC) gets the abstracted network resource information, and shields the networks resources details to Customer Network Controller (CNC).

How to report alarm between PNC and MDSC or between MDSC and CNC is related to the abstraction policy. In figure 1, several different abstraction particles are listed as follow.

1) The NE1, NE2 and NE3 in PNC1 is abstracted as NE1’, the NE4, NE5 and NE6 in PNC1 is abstracted as NE2’ in MDSC.

2) The NE7 and NE8 managed by PNC2 is abstracted as NE3’ in MDSC.

3) The multiple links between NE2 and NE4, which is abstracted as a link between NE1’ and NE2’, uses a link bundling mechanism, and the bandwidth of this link in MDSC is bound to the sum of bandwidth of bundling link in PNC1.

4) MDSC can shield the network resources details, and only provides the information of the edge node and the connection relationship between these nodes.

```
     +-------------------------------------------------------+
     |   +-----+                                     +-----+ |
     |   |NE1''|-------------------------------------|NE2''| |
     |   +-----+                                     +-----+ |
     |CNC  /|\                                         /|\   |
     +------|-----------------------------------------|----+
            |                                         |
     +------|-----------------------------------------|----+
     |   +----+                +----+                +----+  |
     |   |NE1'|----------------|NE2'|----------------|NE3'|  |
     |   |    |---------X------|    |                |    |  |
     |   +----+                +----+                +----+  |
     |MDSC /|\                   /|\                   /|\   |
     +------|---------------------|---------------------|----+
            |                     |                     |
   +-----------------------------------------+         |
   | +-----------------+ +--------------+ PNC1|         |
   | | +---+     +---+ | | +---+        |     |         |
   | | |NE1|--X--|NE2|=====|NE4|----+   |     |         |
   | | +---+     +---+ | | +---+    |   |     | +---------------+
   | |   |         /   | |          |   |     | |          PNC2 |
   | |   |     +---+   | | +---+  +---+ |     | | +---+   +---+ |
   | |   +-----|NE3|---X---|NE5|--|NE6|-----------|NE7|---|NE8| |
   | |         +---+   | | +---+  +---+ |     | | +---+   +---+ |
   | +-----------------+ +--------------+     | |               |
   +-----------------------------------------+ +---------------+
```

Figure 1 Error Report in ACTN architecture

Based on the different abstract methods, when the underlying network resource error occurs, the requirements of alarm reporting mechanism is also different.

1) PNC is able to collect all of the underlying network resource and alarm information.

2) When an error occurs in the link between the NE1 and NE2 in PNC1, it shouldn't report an alarm to MDSC, but report the network resource status has changed inner the NE1', such as the connectivity between the port of NE1', or the maximum available bandwidth has changed, etc..

3) When one of a link between the NE2 and NE4 failures, PNC1 should report the bundling link bandwidth changes to MDSC.

4) When the link between NE3 and NE5 failures, it is abstracted as a link in MDSC, and the PNC1 should report the alarm information, indicating that the link is faulty.

5) In addition, when the underlying network resources failure, due to the abstract policy, PNC reporting the resource status changes to MDSC, such as abstract node internal state changes and abstract link bandwidth property changes, when the upper controller MDSC or CNC received these state changes, it cannot correlation the state changes to the network connections which are impacted. Therefore, the underlying controller PNC or MDSC should report the impact of the LSP and alarm information, the upper layer controller based on these information and correlated with the connection it stored.

3. Abstract for alarm report in ACTN architecture

3.1. Status changes report inner abstract node

TBD

3.2. Status changes report inner abstract link

When an error occurs in a bundling link, the following information should be reported.

- Domain ID;

- Bundling link ID;

- Available Bandwidth;

- Reason for status change;

- Occurrence time.

3.3. Alarm report for abstract node and link

When an error occurs out of a bundling link or for an abstract node, follow alarm information should be reported.

- Domain ID;

- Abstract node ID or abstract link ID

- Alarm reason;

- Alarm level;

- Occurrence time.

3.4. Abstract alarm report for connection

When an error occurs and it impacts a network connection, the follow alarm information should be reported.

- Domain ID;

- Abstract network connection ID;

- Alarm reason;
- Alarm level;

- Occurrence time.

4. Security Considerations

This document raises no new security issues.

5. IANA Considerations

No new IANA considerations are raised by this document.

6. References

6.1. Informative References

[ACTN-Frame] D. Ceccarelli, et al., "Framework for Abstraction and Control of Transport Networks", draft-ceccarelli-actn-framework, work in progress.

Authors' Address

Yunbin Xu
China Academy of Telecom Research
NO.52 Huayuan Beilu, Haidian District, Beijing, China
Email: xuyunbin@catr.cn

Guoying Zhang
China Academy of Telecom Research
NO.52 Huayuan Beilu, Haidian District, Beijing, China
Email: zhangguoying@catr.cn