TEAS Working Group
Internet Draft
Intended status: Informational
Expires: November 2016

Daniele Ceccarelli (Ed)
Ericsson
Young Lee (Ed)
Huawei

April 14, 2016

Framework for Abstraction and Control of Traffic Engineered Networks

draft-ceccarelli-teas-actn-framework-02

Status of this Memo

Copyright Notice

Abstract

Traffic Engineered networks have a variety of mechanisms to facilitate the
separation of the data plane and control plane. They also have a range of management and provisioning protocols to configure and activate network resources. These mechanisms represent key technologies for enabling flexible and dynamic networking.

Abstraction of network resources is a technique that can be applied to a single network domain or across multiple domains to create a single virtualized network that is under the control of a network operator or the customer of the operator that actually owns the network resources.

This draft provides a framework for Abstraction and Control of Traffic Engineered Networks (ACTN).

Table of Contents

1. Introduction

Traffic Engineered 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.

The term Traffic Engineered Network in this draft refers to any connection-oriented network that has the ability of dynamic provisioning, abstracting and orchestrating network resource to the network's clients. Some examples of networks that are in scope of this definition are optical networks, MPLS Transport Profile (MPLS-TP), MPLS Traffic Engineering (MPLS-TE), and other emerging technologies with connection-oriented behavior.

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 compared with 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: Detach the network and service control from underlying technology and help customer express the network as desired by business needs.

- Coordination of resources across multiple domains and multiple layers to provide end-to-end services regardless of whether the domains use SDN or not.

As networks evolve, the need to provide resource 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 Traffic Engineered 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 refers to allowing the customers of network operators (see Section 2.1) to utilize a certain amount of network resources as if they own them and thus control their allocated resources with higher layer or application processes that enables the resources to be used in the most optimal way. 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 a specific service, the set of consumed resources or to a particular customer. Customer controller of the virtual network is envisioned to support a plethora of distinct applications. This means that there may be a further level of virtualization that provides a view of resources in the customer’s virtual network for use by an individual application.

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

- Abstraction of the underlying network resources to higher-layer applications and customers [TE-INFO].

- Virtualization of particular underlying resources, whose selection criterion is the allocation of those resources to a particular customer, application or service. [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-domain networks into a single virtualized network;

- Possibility of providing a customer with virtualized network or 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 presentation of the networks as a virtualized topology to the customers via open and programmable interfaces. This allows for the recursion of controllers in a customer-provider relationship.

1.1. Terminology

The following terms are used in this document. Some of them are newly defined, some others reference existing definition:

- Node: A node is a topological entity describing the "opaque" forwarding aspect of the topological component which represents the opportunity to enable forwarding between points at the edge of the node. It provides the context for instructing the formation, adjustment and removal of the forwarding. A node, in a VN network, can be represented by single physical entity or by a group of nodes moving from physical to virtual network.

- Link: A link is a topological entity describing the effective adjacency between two or more forwarding entities, such as two or more nodes. In its basic form (i.e., point-to-point Link) it associates an edge point of a node with an equivalent edge point on another node. Links in virtual network is in fact connectivity, realized by bandwidth engineering between any two nodes meeting certain criteria, for example, redundancy, protection, latency, not tied to any technology specific characteristics like timeslots or wavelengths. The link can be dynamic, realized by a service in underlay, or static.

- PNC domain: A PNC domain includes all the resources under the control of a single PNC. It can be composed by different routing domains, administrative domains and different layers. The interconnection between PNC domains can be a link or a node.

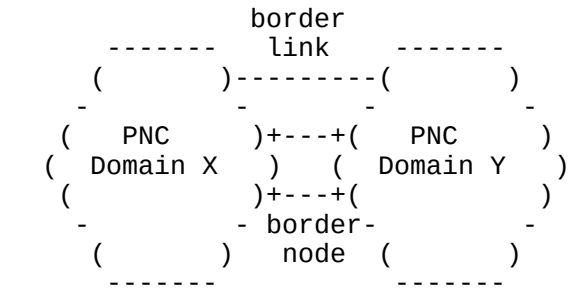


Figure 1 : PNC domain borders

- Virtual Network: A Virtual Network (VN) is a customer view of the transport network. It is composed by a set of physical resources sliced in the provider network and presented to the customer as a set of abstract resources i.e. virtual nodes and virtual links. Depending on the agreement between customer and provider a VN can be just represented by:

  - o How the end points can be connected with given SLA attributes(e.g., re satisfying the customer's objectives)
  - o A pre-configured set of physical resources
  - o Or as outcome of a dynamic request from customer.

  In the first case the VN can be seen at customer level as an e2e connectivity that can be formed by recursive aggregation of lower layers tunnels within the provider domain.
  When the VN is pre-configured, it is provided after a static negotiation between customer and provider while in the third case VN can be dynamically created, deleted, or modified in response to requests from the customer. This implies dynamic changes of network resources reserved for the customer.
  In the second and third case , once that customer has obtained his VN, can act upon the virtual network resources to perform connection management (set-up/release/modify connections).

- Abstract Topology: Every lower controller in the provider network, when is representing its network topology to an higher layer, it may want to hide details of the actual network topology. In such case, an abstract topology may be used for this purpose. Abstract topology enhances scalability for the MDSC to operate multi-domain networks

- Access link: A link between a customer node and a provider node.

- Inter domain link: A link between domains managed by different PNCs. The MDSC is in charge of managing inter-domain links.

- Border node: A node whose interfaces belong to different domains. It may be managed by different PNCs or by the MDSC.

- Access Point (AP): An access point is defined on an access link. It is used to keep confidentiality between the customer and the provider. It is an identifier shared between the customer and the provider, used to map the end points of the border node in the provider NW. The AP can be used by the customer when requesting connectivity service to the provider. A number of parameters, e.g. available bandwidth, need to be associated to the AP to qualify it.

- VN Access Point (VNAP): A VNAP is defined within an AP as part of a given VN and is used to identify the portion of the AP, and hence of the access link) dedicated to a given VN.

2. Business Model of ACTN

The Virtual Private Network (VPN) [RFC4026] and Overlay Network (ON) models [RFC4208] are built on the premise that one single network provider provides all virtual private or overlay networks to its customers. These models are simple to operate but have 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 service offering is not enough and it is desirable to provide each of them with a customized virtual network service.

Advanced customers may own dedicated virtual resources, or share resources. They may also have the ability to modify their service parameters within the scope of their virtualized environments.

As customers are geographically spread over multiple network provider domains, they have to interface multiple providers and may have to support multiple virtual network services with different underlying objectives set by the network providers. To enable these customers to support flexible and dynamic applications they need to control their allocated virtual network resources in a dynamic fashion, and that means that they need an abstracted view of the topology that spans all of the network providers.

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

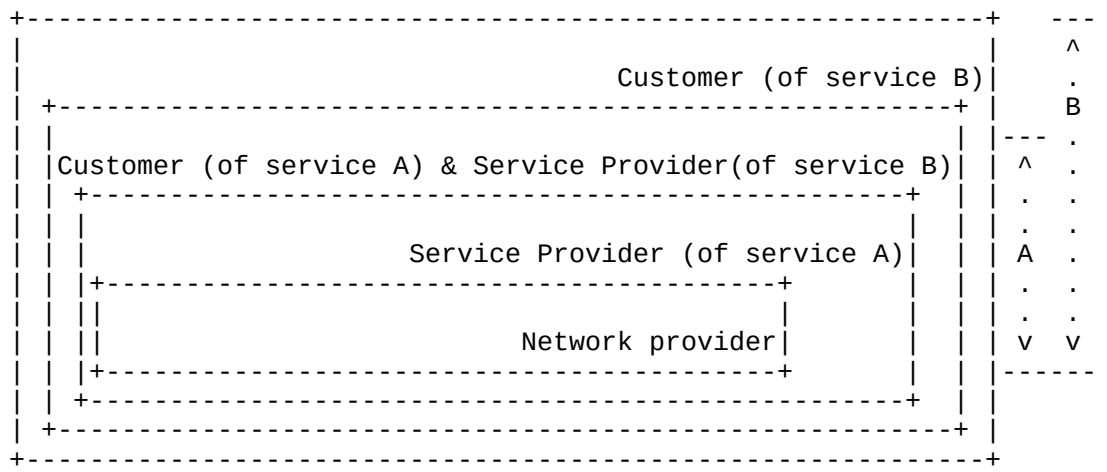


Figure 2 : Service Recursiveness.

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 as shown in Figure 2 where Service Provider A uses resources from Network Provider A and Network Provider B to offer a virtualized network to its customer.

Figure 3 : A service Provider as Customer of Two Network Providers.

The ACTN network model is predicated upon this three tier model and is summarized in Figure 3:

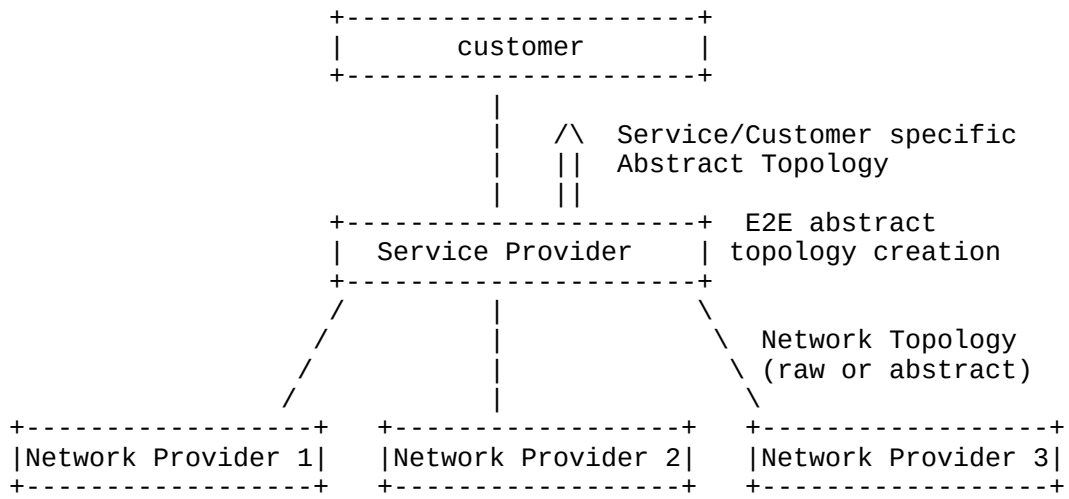


Figure 4 : Three tier model.

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 customers, 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.

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 virtual services that are coordinated under ACTN can be split into two categories:

. 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 in NFV (e.g. cloud) providers and require connectivity between a customer site and the NFV provider site (e.g. a data center). These VNF services may include a security function like firewall, a traffic optimizer, the provisioning of storage or computation capacity. In these cases the customer does not care whether the service is implemented in a given data center or another. This allows the network provider divert customer requests where most suitable. This is also known as "end points mobility" case. (See Section 4.2.3)

The types of controller defined are shown in Figure 4 below and are the following:

. CNC - Customer Network Controller
. MDSC - Multi Domain Service Coordinator

. PNC - Physical Network Controller

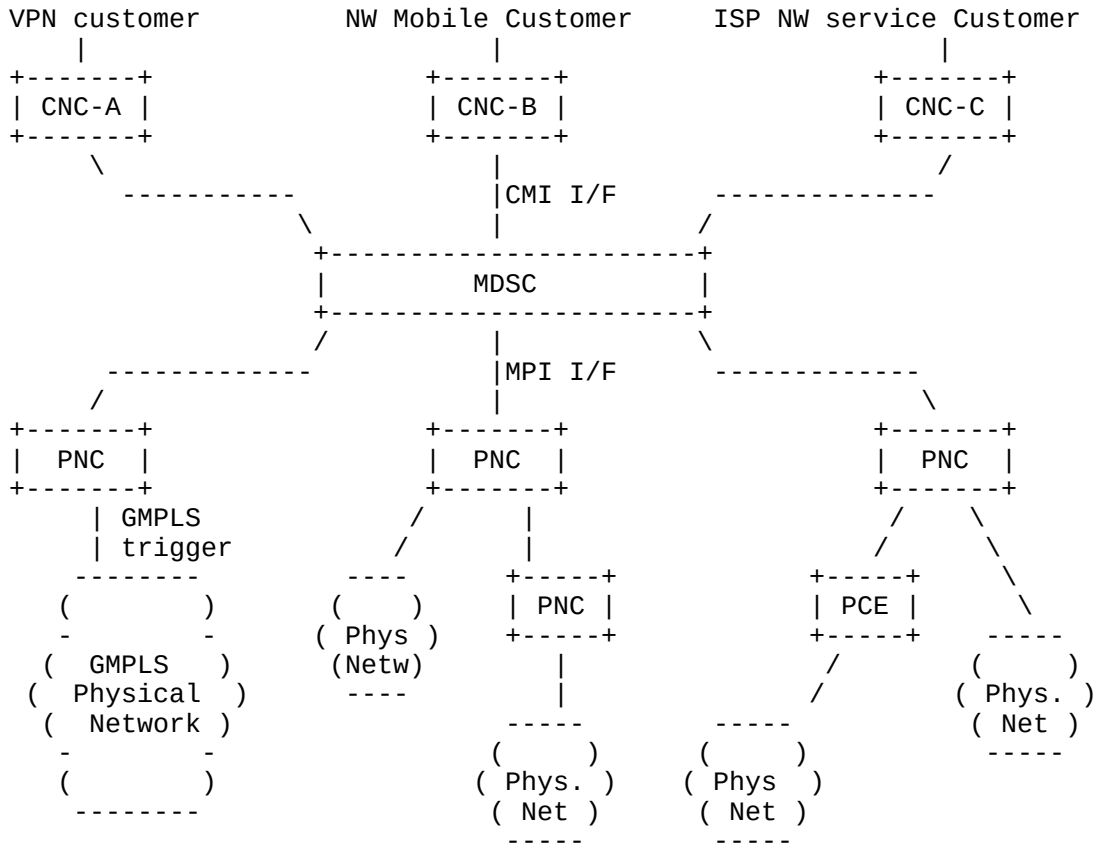


Figure 5 : ACTN Control Hierarchy

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 applications, 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.

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. The key point of the MDSC and the whole ACTN framework is detaching the network and service control from underlying technology and help customer express the network as desired by business needs. The MDSC envelopes the instantiation of right technology and network control to meet business criteria. In essence it controls and manages the primitives to achieve functionalities as desired by CNC
A hierarchy of MDSCs can be foreseen for scalability and administrative choices.

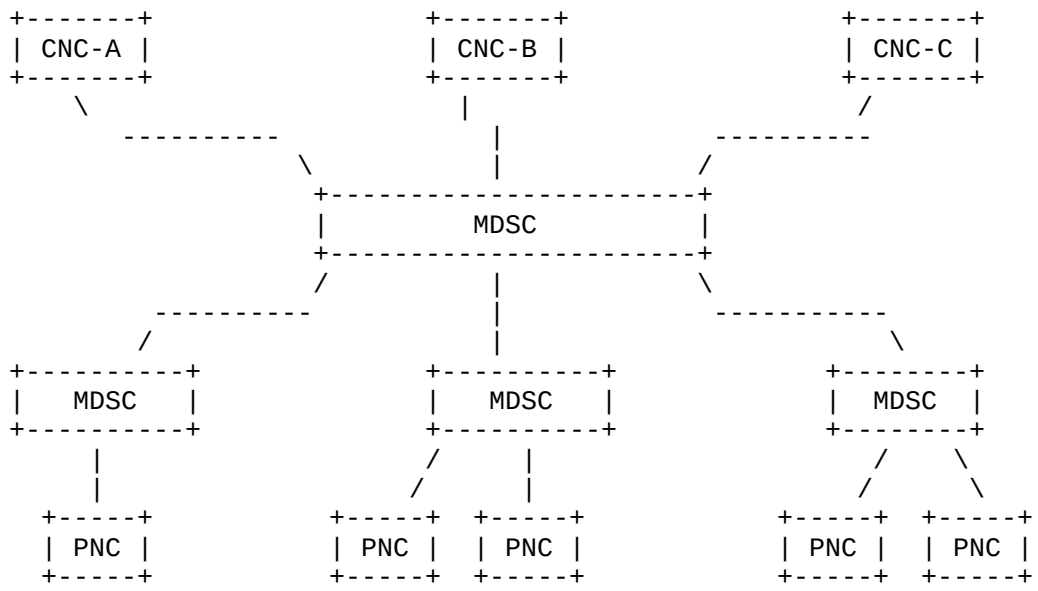


Figure 6 : Controller recursiveness

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).

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.

Figure 5 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 5 (ACTN Interfaces) below:

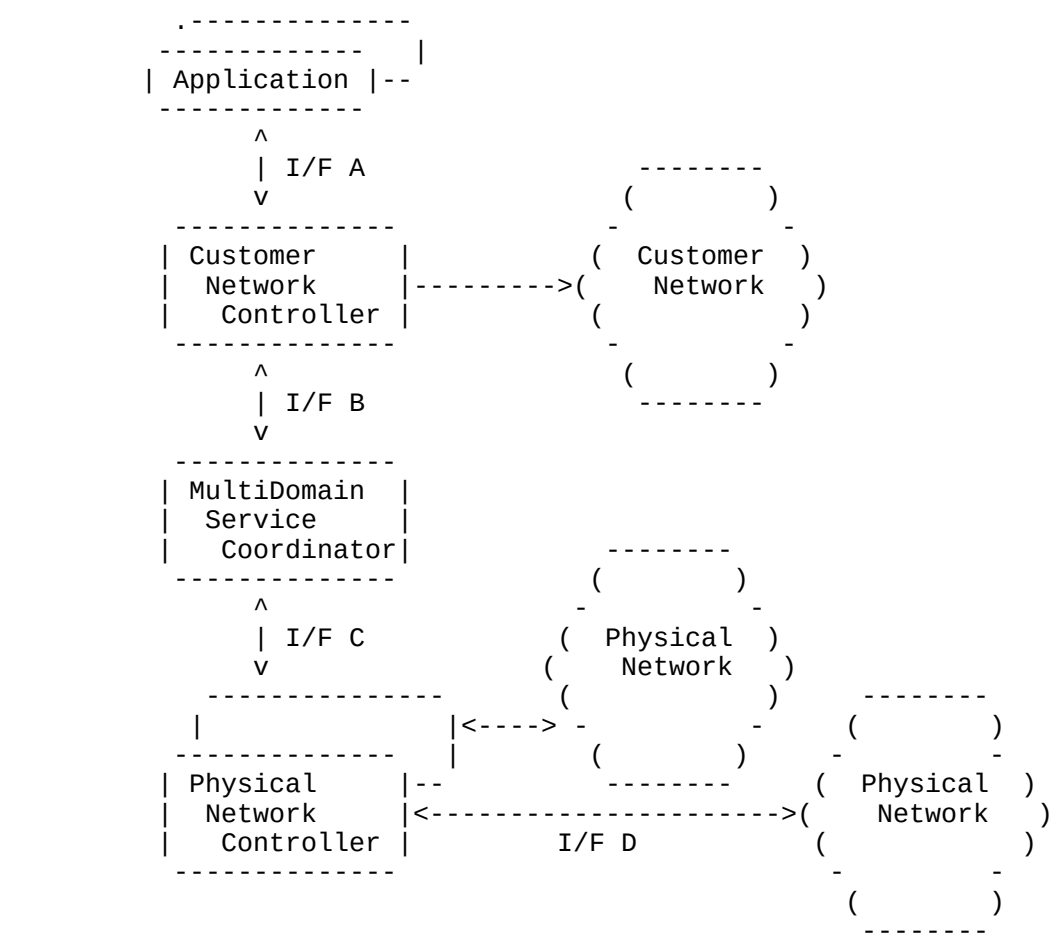


Figure 7 : ACTN Interfaces

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.

The interfaces within the ACTN scope are B and C.

4. VN creation process

The provider can present to the customer different level of network abstraction, spanning from one extreme (say "black") where nothing is shown, just the APs, to the other extreme (say "white") where a slice of the network is shown to the customer. There are shades of gray in between where a number of abstract links and nodes can be shown.

The VN creation is composed by two phases: Negotiation and Implementation.

Negotiation: In the case of grey/white topology abstraction, there is an a priori phase in which the customer agrees with the provider on the type of topology to be shown, e.q. 10 virtual links and 5 virtual nodes with a given interconnectivity. This is something that is assumed to be preconfigured by the operator off-line, what is online is the capability of modifying/deleting something (e.g. a virtual link). In the case of "black" abstraction this negotiation phase does not happen, in the sense that the customer can only see the APs of the network.

Implementation: In the case of black topology abstraction, the customers can ask for connectivity with given constraints/SLA

between the APs and LSPs/tunnels are created by the provider to satisfy the request. What the customer sees is only that his CEs are connected with a given SLA. In the case of grey/white topology the customer creates his own LSPs accordingly to the topology that was presented to him.

5. Access Points and Virtual Network Access Points

In order not to share unwanted topological information between the customer domain and provider domain, a new entity is defined and associated to an access link, the Access Point (AP). See the definition of AP in Section 1.1.

A customer node will use APs as the end points for the request of VNs.

A number of parameters need to be associated to the APs. Such parameters need to include at least: the maximum reservable bandwidth on the link, the available bandwidth and the link characteristics (e.g. switching capability, type of mapping).

Editor note: it is not appropriate to define link characteristics like bandwidth against a point (AP). A solution needs to be found.

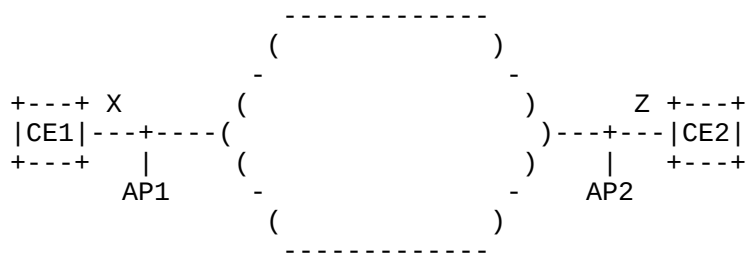


Figure 8 : APs definition customer view

Let’s take as example a scenario in which CE1 is connected to the network via a 10Gb link and CE2 via a 40Gb link. Before the creation of any VN between AP1 and AP2 the customer view can be summarized as follows:

AP id	MaxResBw	AvailableBw	CE,port
AP1	10Gb	10Gb	CE1,portX
AP2	40Gb	40Gb	CE2,portZ

Table 1: AP - customer view

On the other side what the provider sees is:

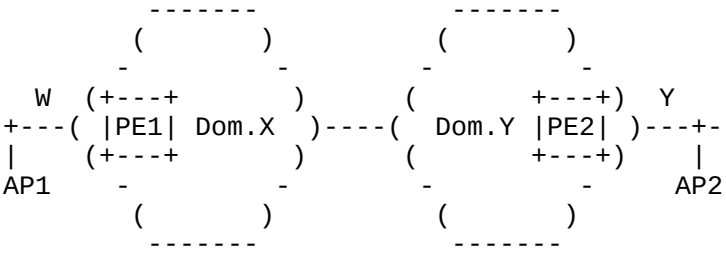


Figure 9 : Provider view of the AP

Which in the example above ends up in a summarization as follows:

AP id	MaxResBw	AvailableBw	PE,port
AP1	10Gb	10Gb	PE1,portW
AP2	40Gb	40Gb	PE2,portY

Table 2: AP - provider view

The second entity that needs to be defined is a structure within the AP that is linked to a VN and that is used to allow for different VN to be provided starting from the same AP. It also allows reserving the bandwidth for the VN on the access link. Such entity is called Virtual Network Access Point. For each virtual network is defined on an AP, a different VNAP is created.

In the simple scenario depicted above we suppose to create two virtual networks. The first one has with VN identifier 9 between AP1

and AP2 with and bandwidth of 1Gbps, while the second one with VN id 5, again between AP1 and AP2 and bandwidth 2Gbps.

The customer view would evolve as follows:

AP/VNAPid	MaxResBw	AvailableBw	PE,port
AP1	10Gbps	7Gbps	PE1,portW
-VNAP1.9	1Gbps	N.A.	
-VNAP1.5	2Gbps	N.A	
AP2	40Gb	37Gb	PE2,portY
-VNAP2.9	1Gbps	N.A.	
-VNAP2.5	2Gbps	N.A	

Table 3: AP and VNAP - provider view after VN creation

5.1. Dual homing scenario

Often there is a dual homing relationship between a CE and a pair of PE. This case needs to be supported also by the definition of VN, AP and VNAP. Suppose to have CE1 connected to two different PE in the operator domain via AP1 and AP2 and the customer needing 5Gbps of bandwidth between CE1 and CE2.

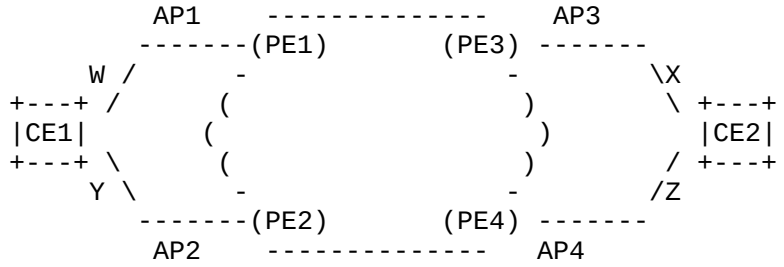


Figure 10 : Dual homing scenario

In this case the customer will request for a VN between AP1, AP2 and AP3 specifying a dual homing relationship between AP1 and AP2. As a consequence no traffic will be flowing between AP1 and AP2. The dual homing relationship would then be mapped against the VNAPs (since other independent VNs might have AP1 and AP2 as end points).

The customer view would be as follows:

AP/VNAPid	MaxResBw	AvailableBw	CE,port	Dual Homing
AP1 -VNAP1.9	10Gbps 5Gbps	5Gbps N.A.	CE1,portW	VNAP2.9
AP2 -VNAP2.9	40Gbps 5Gbps	35Gbps N.A.	CE1,portY	VNAP1.9
AP3 -VNAP3.9	40Gbps 5Gbps	35Gbps N.A.	CE2,portZ	NONE

Table 4: Dual homing - customer view after VN creation

6. End point selection & mobility

Virtual networks could be used as the infrastructure to connect a number of sites of a customer among them or to provide connectivity between customer sites and virtualized network functions (VNF) like for example virtualized firewall, vBNG, storage, computational functions.

6.1. End point selection & mobility

A VNF could be deployed in different places (e.g. data centers A, B or C in figure below) but the VNF provider (=ACTN customer) doesn’t know which is the best site where to install the VNF from a network point of view (e.g. latency). For example it is possible to compute the path minimizing the delay between AP1 and AP2, but the customer doesn’t know a priori if the path with minimum delay is towards A, B or C.

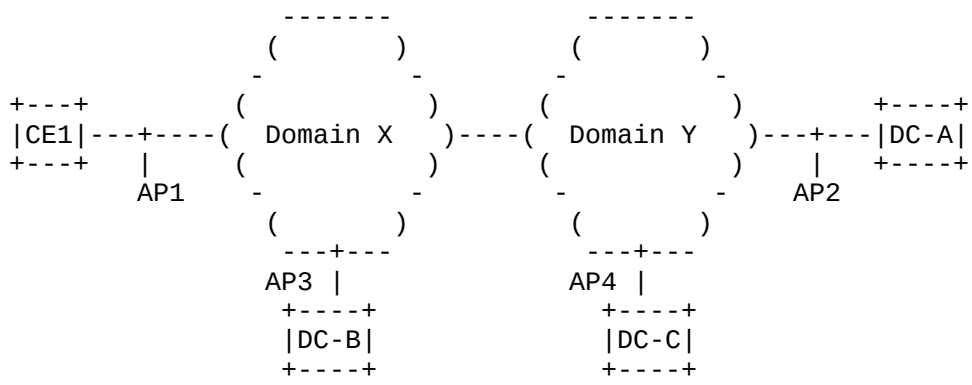


Figure 11 : End point selection

In this case the VNF provider (=ACTN customer) should be allowed to ask for a VN between AP1 and a set of end points. The list of end points is provided by the VNF provider. When the end point is identified the connectivity can be instantiated and a notification can be sent to the VNF provider for the instantiation of the VNF.

6.2. Preplanned end point migration

A premium SLA for VNF service provisioning consists on the offering of a protected VNF instantiated on two or more sites and with a hot stand-by protection mechanism. In this case the VN should be provided so to switch from one end point to another upon a trigger from the VNF provider or an automatic failure detection mechanism. An example is provided in figure below where the request from the VNF provider is for connectivity with given constraint and resiliency between CE1 and a VNF with primary installation in DC-A and a protection in DC-C.

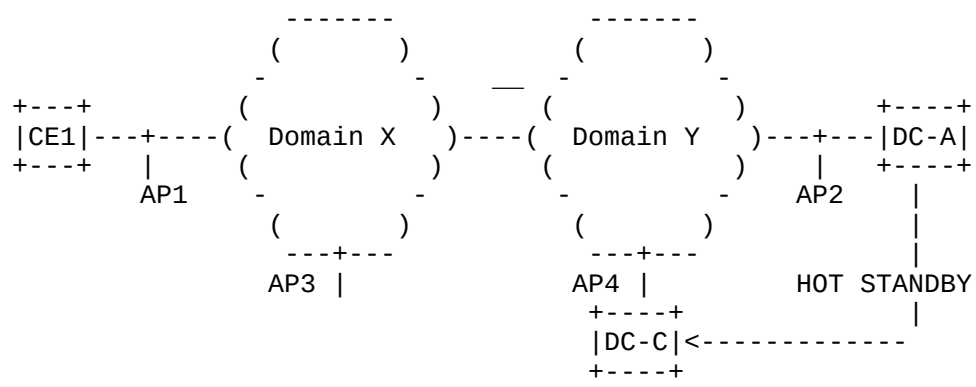


Figure 12 : Preplanned endpoint migration

6.3. On the fly end point migration

The one the fly end point migration concept is very similar to the end point selection one. The idea is to give the provider not only the list of sites where the VNF can be installed, but also a mechanism to notify changes in the network that have impacts on the SLA. After an handshake with the customer controller/applications, the bandwidth in network would be moved accordingly with the moving of the VNFs.

7. Security

TBD

8. References

8.1. Informative References

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

[RFC4026] L. Andersson, T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, March 2005.

[RFC4208] G. Swallow, J. Drake, H.Ishimatsu, Y. Rekhter, "Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model", RFC 4208, October 2005.

[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.

[TE-INFO] A. Farrel, Editor, "Problem Statement and Architecture for Information Exchange Between Interconnected Traffic Engineered Networks", draft-ietf-teas-interconnected-te-info-exchange, work in progress.

[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.

9. 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

Gert Grammel
Juniper Networks
ggrammel@juniper.net

TEAS Working Group
Internet-Draft
Intended Status: Standards Track
Expires: September 11, 2017

R. Gandhi, Ed.
Cisco Systems, Inc.
H. Shah
Ciena
Jeremy Whittaker
Verizon

March 10, 2017

Fast Reroute Procedures For Associated Bidirectional Label Switched Paths (LSPs)
draft-gandhishah-teas-assoc-corouted-bidir-04

Abstract

Resource Reservation Protocol (RSVP) association signaling can be used to bind two unidirectional LSPs into an associated bidirectional LSP. When an associated bidirectional LSP is co-routed, the reverse LSP follows the same path as its forward LSP. This document describes Fast Reroute (FRR) procedures for both single-sided and double-sided provisioned associated bidirectional LSPs. The FRR procedures are applicable to co-routed and non co-routed LSPs. For co-routed LSPs, the FRR procedures can ensure that traffic flows on co-routed paths in the forward and reverse directions after a failure event.

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."

Table of Contents

1. Introduction

The Resource Reservation Protocol (RSVP) (Extended) ASSOCIATION Object is specified in [RFC6780] which can be used generically to associate (G)Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs). [RFC7551] defines mechanisms for binding two point-to-point unidirectional LSPs [RFC3209] into an associated bidirectional LSP. There are two models described in [RFC7551] for provisioning an associated bidirectional LSP, single-sided and double-sided. In both models, the reverse LSP of the bidirectional LSP may or may not be co-routed and follow the same path as its forward LSP.

[GMPLS-FRR] defines Fast Reroute (FRR) procedure for GMPLS signaled LSPs to co-ordinate bypass tunnel assignments in the forward and reverse directions. The mechanisms defined in [GMPLS-FRR] are applicable to FRR of associated bidirectional LSPs.

In packet transport networks, there are requirements where the reverse LSP of a bidirectional LSP needs to follow the same path as its forward LSP [RFC6373]. The MPLS Transport Profile (TP) [RFC6370] architecture facilitates the co-routed bidirectional LSP by using the GMPLS extensions [RFC3473] to achieve congruent paths. However, the RSVP association signaling allows to enable co-routed bidirectional LSPs without having to deploy GMPLS extensions in the existing networks. The association signaling also allows to take advantage of the existing Traffic Engineering (TE) and FRR mechanisms in the network.

This document describes FRR procedures for both single-sided and double-sided provisioned associated bidirectional LSPs. The FRR procedures are applicable to co-routed and non co-routed LSPs. For co-routed LSPs, the FRR procedures can ensure that traffic flows on co-routed paths in the forward and reverse directions after a failure event.

2. Conventions Used in This Document

2.1. Key Word Definitions

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].

2.2. Terminology

The reader is assumed to be familiar with the terminology in [RFC2205], [RFC3209], [RFC4090] and [RFC7551].

2.2.1. Reverse Co-routed Unidirectional LSPs

Two reverse unidirectional point-to-point (P2P) LSPs are setup in the opposite directions between a pair of source and destination nodes to form an associated bidirectional LSP. A reverse unidirectional LSP originates on the same node where the forward unidirectional LSP terminates, and it terminates on the same node where the forward unidirectional LSP originates. A reverse co-routed unidirectional LSP traverses along the same path of the forward direction unidirectional LSP in the opposite direction.

3. Overview

As specified in [RFC7551], in the single-sided provisioning case, the RSVP TE tunnel is configured only on one endpoint node of the bidirectional LSP. An LSP for this tunnel is initiated by the originating endpoint with (Extended) ASSOCIATION Object containing Association Type set to "single-sided associated bidirectional LSP" and REVERSE_LSP Object inserted in the Path message. The remote endpoint then creates the corresponding reverse TE tunnel and signals the reverse LSP in response using the information from the REVERSE_LSP Object and other objects present in the received Path message. As specified in [RFC7551], in the double-sided provisioning case, the RSVP TE tunnel is configured on both endpoint nodes of the bidirectional LSP. Both forward and reverse LSPs are initiated independently by the two endpoints with (Extended) ASSOCIATION Object containing Association Type set to "double-sided associated bidirectional LSP". In both single-sided and double-sided provisioned bidirectional LSPs, the reverse LSP may or may not be congruent (i.e. co-routed) and follow the same path as its forward LSP.

In the case of single-sided provisioned LSP, the originating LSP with REVERSE_LSP Object is identified as a forward LSP. In the case of double-sided provisioned LSP, the LSP originating from the higher node address (as source) and terminating on the lower node address (as destination) is identified as a forward LSP. The reverse LSP of the bidirectional LSP traverses in the opposite direction of the forward LSP.

Both single-sided and double-sided associated bidirectional LSPs require solutions to the following issues for fast reroute.

3.1. Fast Reroute Bypass Tunnel Assignment

In order to ensure that the traffic flows on a co-routed path after a link or node failure on the protected LSP path, the mid-point Point

of Local Repair (PLR) nodes need to assign matching bidirectional bypass tunnels for fast reroute. Even for a non co-routed bidirectional LSP, it is desired that the same bidirectional bypass tunnel is used in both directions of the protected LSP. Such bypass assignment requires co-ordination between the forward and reverse direction PLR nodes when more than one bypass tunnels are present on a PLR node.

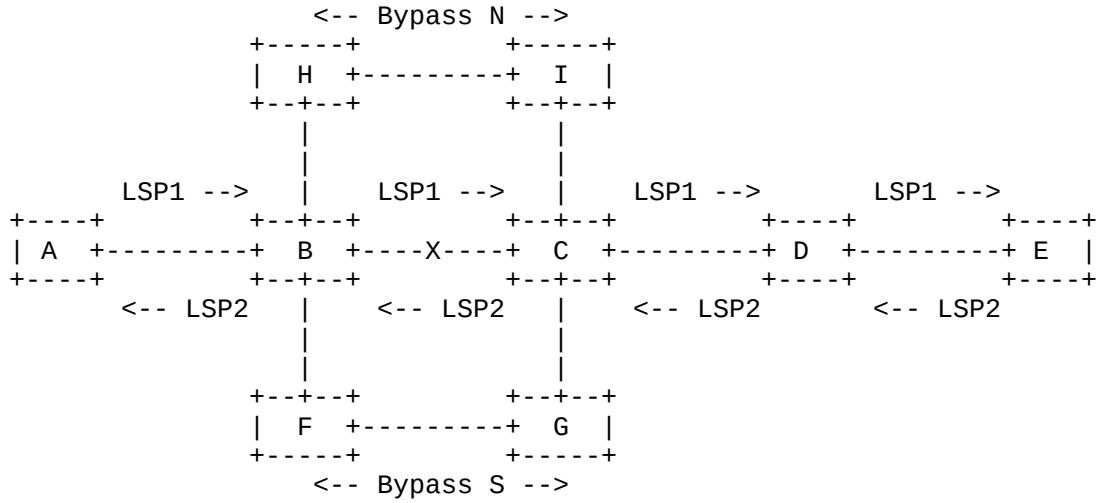


Figure 1: Multiple Bidirectional Bypass Tunnels

As shown in Figure 1, there are two bypass tunnels available, Bypass N on path B-H-I-C and Bypass S on path B-F-G-C. The mid-point PLR nodes B and C need to co-ordinate bypass tunnel assignment to ensure that traffic in both directions flow through either on the Bypass N path B-H-I-C or the Bypass S path B-F-G-C, after the link B-C failure.

3.2. Bidirectional LSP Association At Mid-Points

In packet transport networks, a restoration LSP is signaled after a link failure on the protected LSP and the protected LSP may or may not be torn down [GMPLS-REST]. In this case, multiple forward and reverse LSPs of a bidirectional LSP may be present at mid-point nodes with identical (Extended) ASSOCIATION Objects. This creates an ambiguity at mid-point nodes to identify the correct associated LSP pair for fast reroute bypass assignment (e.g. during the recovery phase of RSVP graceful restart procedure).

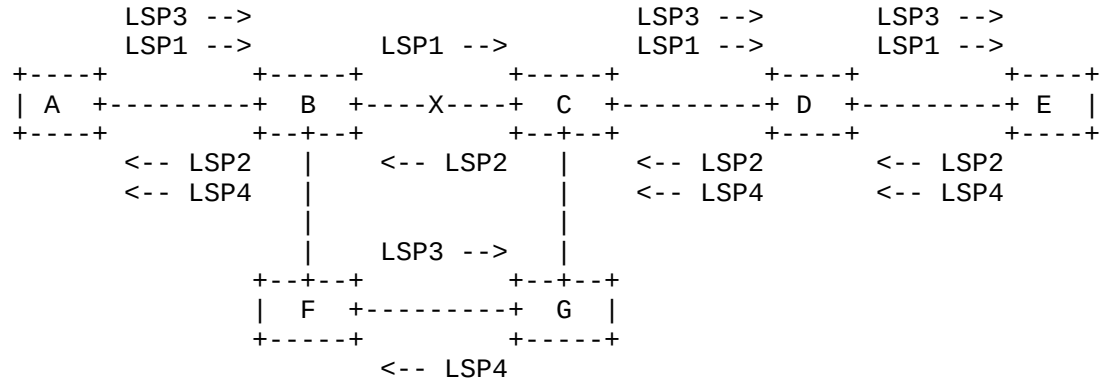


Figure 2: Restoration LSP Set-up After Link Failure

As shown in Figure 2, protected LSPs LSP1 and LSP2 are an associated LSP pair, similarly restoration LSPs LSP3 and LSP4 are an associated LSP pair, both pairs belong to the same associated bidirectional LSP and carry identical (Extended) ASSOCIATION Objects. In this example, mid-point node D may mistakenly associate LSP1 with reverse LSP4 instead of reverse LSP3 due to the matching (Extended) ASSOCIATION Objects. This may cause the bidirectional LSP to become non co-routed. Since a reverse LSP reflects the bypass tunnel assignment received in the forward LSP, this can also lead to undesired bypass tunnel assignments.

4. Signaling Procedure

4.1. Bidirectional LSP Fast Reroute

The mechanisms defined in [GMPLS-FRR] are used for fast reroute of both single-sided and double-sided associated bidirectional LSPs as following.

- As described in [GMPLS-FRR], BYPASS_ASSIGNMENT subobject is signaled in the RRO of the Path message to co-ordinate bypass tunnel assignment between the forward and reverse direction PLR nodes. A BYPASS_ASSIGNMENT subobject MUST be added by the forward direction PLR node in the Path message of the forward LSP to indicate the bypass tunnel assigned.

- The forward direction PLR node always initiates the bypass tunnel assignment for the forward LSP. The reverse direction PLR (forward direction LSP Merge Point (MP)) node simply reflects the bypass tunnel assignment for the reverse direction LSP.

- o After a link or node failure, the PLR nodes in both forward and reverse directions trigger fast reroute independently using the procedures defined in [RFC4090].

- o When using a node protection bypass tunnel, asymmetry of paths can occur in the forward and reverse directions of the bidirectional LSP after a link failure when using co-routed LSPs [GMPLS-FRR]. This can be corrected using the re-corouting procedure defined in [GMPLS-FRR]. Unlike GMPLS LSPs, the asymmetry of paths in the forward and reverse directions does not result in RSVP soft-state time-out with the associated bidirectional LSPs.

4.2. Bidirectional LSP Association At Mid-points

In order to associate the correct LSPs at a mid-point node, an endpoint node MUST signal Extended ASSOCIATION Object and add unique Extended Association ID for each associated forward and reverse LSP pair forming the bidirectional LSP. As an example, an endpoint node MAY set the Extended Association ID to the value specified in Section 5.1 of this document.

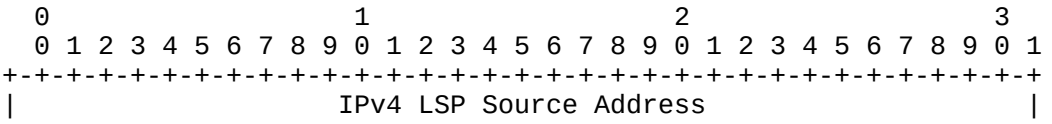
- o For single-sided provisioned bidirectional LSPs [RFC7551], the originating endpoint signals the Extended ASSOCIATION Object with a unique Extended Association ID. The remote endpoint copies the contents of the received Extended ASSOCIATION Object including the Extended Association ID in the RSVP Path message of the reverse LSP's Extended ASSOCIATION Object.

- o For double-sided provisioned bidirectional LSPs [RFC7551], both endpoints need to ensure that the bidirectional LSP has a unique Extended ASSOCIATION Object for each forward and reverse LSP pair by provisioning appropriate Extended Association IDs signaled by them.

5. Message and Object Definitions

5.1. Extended ASSOCIATION Object

The Extended Association ID in the Extended ASSOCIATION Object can be set to the value specified as following to uniquely identify associated forward and reverse LSP pair of a bidirectional LSP.



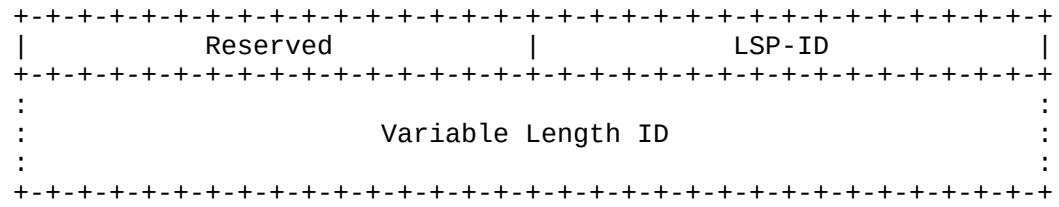


Figure 3: IPv4 Extended Association ID

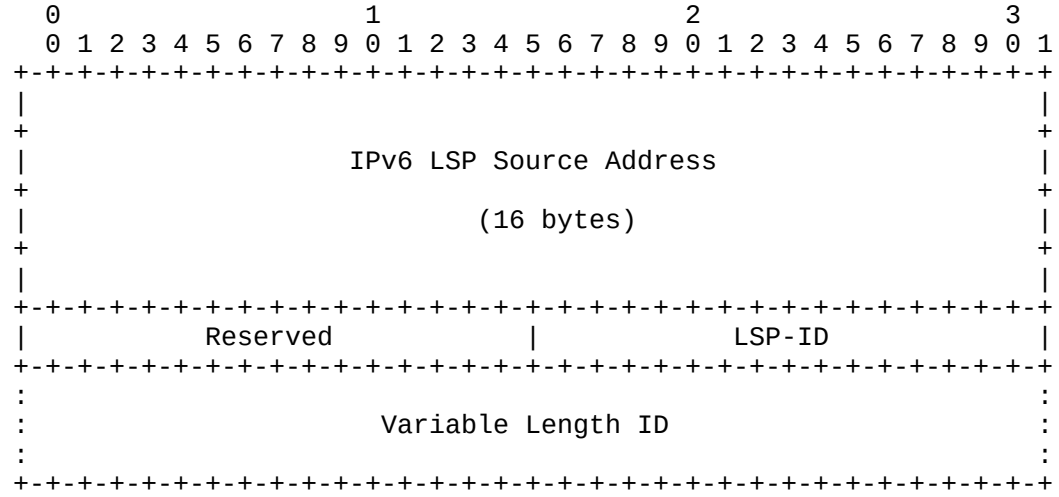


Figure 4: IPv6 Extended Association ID

LSP Source Address

IPv4/IPv6 source address of the forward LSP.

LSP-ID

16-bits LSP-ID of the forward LSP.

Variable Length ID

Variable length ID inserted by the endpoint node of the associated bidirectional LSP [RFC6780].

6. Compatibility

This document describes the procedures for fast reroute for associated bidirectional LSPs. Operators wishing to use this function SHOULD ensure that it is supported on the nodes on the LSP path.

7. Security Considerations

This document uses signaling mechanisms defined in [RFC7551] and [GMPLS-FRR] and does not introduce any additional security considerations other than already covered in [RFC7551], [GMPLS-FRR] and the MPLS/GMPLS security framework [RFC5920].

8. IANA Considerations

This document does not make any request for IANA action.

9. References

9.1. Normative References

[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.

[RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, May 2005.

[RFC6780] Berger, L., Le Faucheur, F., and A. Narayanan, "RSVP Association Object Extensions", RFC 6780, October 2012.

[RFC7551] Zhang, F., Ed., Jing, R., and Gandhi, R., Ed., "RSVP-TE Extensions for Associated Bidirectional LSPs", RFC 7551, May 2015.

[GMPLS-FRR] Taillon, M., Saad, T., Ed., Gandhi, R., Ed., Ali, Z., Bhatia, M., "Extensions to Resource Reservation Protocol For Fast Reroute of Traffic Engineering GMPLS LSPs", draft-ietf-teas-gmpls-lsp-fastreroute, work in progress.

9.2. Informative References

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

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

[RFC6370] Bocci, M., Swallow, G., and E. Gray, "MPLS Transport Profile (MPLS-TP) Identifiers", RFC 6370, September 2011.

[RFC6373] Andersson, L., Berger, L., Fang, L., Bitar, N., and E. Gray, "MPLS Transport Profile (MPLS-TP) Control Plane Framework", RFC 6373, September 2011.

[GMPLS-REST] Zhang, X., Zheng, H., Ed., Gandhi, R., Ed., Ali, Z.,

Brzozowski, P., "RSVP-TE Signaling Procedure for End-to-End GMPLS Restoration and Resource Sharing", draft-ietf-teas-gmpls-resource-sharing-proc, work in progress.

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.

EMail: rgandhi@cisco.com

Himanshu Shah
Ciena

EMail: hshah@ciena.com

Jeremy Whittaker
Verizon

EMail: jeremy.whittaker@verizon.com

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 6, 2017

T. Saad, Ed.
R. Gandhi, Ed.
Z. Ali
Cisco Systems, Inc.
R. Venator
Defense Information Systems Agency
Y. Kamite
NTT Communications Corporation

February 2, 2017

RSVP Extensions For Re-optimization of Loosely Routed Point-to-Multipoint Traffic Engineering Label Switched Paths (LSPs)
draft-ietf-teas-p2mp-loose-path-reopt-09

Abstract

Re-optimization of a Point-to-Multipoint (P2MP) Traffic Engineered (TE) Label Switched Path (LSP) may be triggered based on the need to re-optimize an individual source-to-leaf (S2L) sub-LSP or a set of S2L sub-LSPs, both using Sub-Group-Based Re-optimization method, or the entire P2MP-TE LSP tree using the Make-Before-Break (MBB) method. This document discusses the application of the existing mechanisms for path re-optimization of loosely routed Point-to-Point (P2P) TE LSPs to the P2MP-TE LSPs, identifies issues in doing so and defines procedures to address them. When re-optimizing a large number of S2L sub-LSPs in a tree using the Sub-Group-Based Re-optimization method, the S2L sub-LSP descriptor list may need to be semantically fragmented. This document defines the notion of a fragment identifier to help recipient nodes unambiguously reconstruct the fragmented S2L sub-LSP descriptor list.

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."

Copyright Notice

Table of Contents

1. Introduction

This document defines Resource Reservation Protocol - Traffic Engineering (RSVP-TE) [RFC2205] [RFC3209] signaling extensions for re-optimizing loosely routed Point-to-Multipoint (P2MP) Traffic Engineered (TE) Label Switched Paths (LSPs) [RFC4875] in a Multi-Protocol Label Switching (MPLS) or Generalized MPLS (GMPLS) [RFC3473] network.

A P2MP-TE LSP is comprised of one or more source-to-leaf (S2L) sub-LSPs. A loosely routed P2MP-TE S2L sub-LSP is defined as one whose path does not contain the full explicit route identifying each node along the path to the egress node at the time of its signaling by the ingress node. Such an S2L sub-LSP is signaled with no Explicit Route Object (ERO) [RFC3209], or with an ERO that contains at least one loose next-hop, or with an ERO that contains an abstract node which identifies more than one node. This is often the case with inter-domain P2MP-TE LSPs where Path Computation Element (PCE) is not used [RFC5440].

As per [RFC4875], an ingress node may re-optimize the entire P2MP-TE LSP tree by re-signaling all its S2L sub-LSP(s) using the Make-Before-Break (MBB) method or may re-optimize individual S2L sub-LSP or a set of S2L sub-LSPs i.e. individual destination or a set of destinations, both using the Sub-Group-Based Re-optimization method.

[RFC4736] defines RSVP signaling procedure for re-optimizing the path(s) of loosely routed Point-to-Point (P2P) TE LSP(s). Those mechanisms include a method for the ingress node to trigger a new path re-evaluation request and a method for the mid-point node to notify availability of a preferred path. This document discusses the application of those mechanisms to the re-optimization of loosely routed P2MP-TE LSPs, identifies issues in doing so and defines procedures to address them.

For re-optimizing a group of S2L sub-LSPs in a tree using the Sub-Group-Based Re-optimization method, an S2L sub-LSP descriptor list can be used to signal one or more S2L sub-LSPs in an RSVP message. This RSVP message may need to be semantically fragmented when large number of S2L sub-LSPs are added to the descriptor list. This document defines the notion of a fragment identifier to help recipient nodes unambiguously reconstruct the fragmented S2L sub-LSP descriptor list.

2. Conventions Used in This Document

2.1. Key Word Definitions

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

2.2. Abbreviations

ABR: Area Border Router.

AS: Autonomous System.

ERO: Explicit Route Object.

LSR: Label Switching Router.

S2L sub-LSP: Source-to-leaf sub Label Switched Path.

TE LSP: Traffic Engineering Label Switched Path.

TE LSP ingress: Head-end/source node of the TE LSP.

TE LSP egress: Tail-end/destination node of the TE LSP.

2.3. Terminology

The reader is assumed to be familiar with the terminology in [RFC4736] and [RFC4875].

3. Overview

[RFC4736] defines RSVP signaling extensions for re-optimizing loosely routed P2P TE LSPs as follows:

- A mid-point LSR that expands loose next-hop(s) sends a solicited or unsolicited PathErr with the Notify error code 25 (as defined in [RFC3209]) with sub-code 6 to indicate "Preferable Path Exists" to the ingress node.

- An ingress node triggers a path re-evaluation request at all mid-point LSR(s) that expands loose next-hop(s) by setting the "Path Re-evaluation Request" flag (0x20) in SESSION_ATTRIBUTES Object in the Path message.

- The ingress node upon receiving this PathErr with the Notify error code either solicited or unsolicited initiates re-optimization of the LSP using the MBB method with a different LSP-ID.

The following sections discuss the issues that may arise when

applying the mechanisms defined in [RFC4736] for re-optimizing loosely routed P2MP-TE LSPs.

3.1. Loosely Routed Inter-domain P2MP-TE LSP Tree

An example of a loosely routed inter-domain P2MP-TE LSP tree is shown in Figure 1. In this example, the P2MP-TE LSP tree consists of 3 S2L sub-LSPs, to destinations (i.e. leafs) R10, R11 and R12 from the ingress node (i.e. source) R1. Nodes R2 and R5 are branch nodes and nodes ABR3, ABR4, ABR7, ABR8 and ABR9 are area border routers. For the S2L sub-LSP to destination R10, nodes ABR3, ABR7 and R10 are defined as loose next-hops. For the S2L sub-LSP to destination R11, nodes ABR3, ABR8 and R11 are defined as loose next-hops. For the S2L sub-LSP to destination R12, nodes ABR4, ABR9 and R12 are defined as loose next-hops.

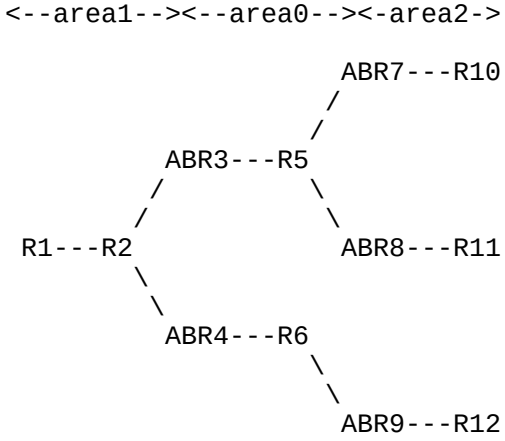


Figure 1: An Example of Loosely Routed Inter-domain P2MP-TE LSP Tree

3.2. Existing Mechanism For Tree-Based P2MP-TE LSP Re-optimization

Mechanisms defined in [RFC4736] can be easily applied to trigger the re-optimization of individual or group of S2L sub-LSP(s). However, to apply these [RFC4736] mechanisms for triggering the re-optimization of a P2MP-TE LSP tree, an ingress node needs to send path re-evaluation requests on all (typically 100s of) S2L sub-LSPs and the mid-point LSR needs to send PathErrs with the Notify error code for all S2L sub-LSPs. Such mechanisms may lead to the following issues:

o A mid-point LSR that expands loose next-hop(s) may have to

accumulate the received path re-evaluation request(s) for all S2L sub-LSPs (e.g. by using a wait timer) and interpret them as a re-optimization request for the whole P2MP-TE LSP tree. Otherwise, a mid-point LSR may prematurely notify "Preferable Path Exists" for one or a sub-set of S2L sub-LSPs.

- o Similarly, the ingress node may have to heuristically determine when to perform P2MP-TE LSP tree re-optimization and when to perform S2L sub-LSP re-optimization. For example, an implementation may choose to delay re-optimization long enough to allow all PathErr(s) to be received. Such timer-based procedures may produce undesired results.

- o The ingress node that receives (un)solicited PathErr(s) with the Notify error code for individual S2L sub-LSP(s), may prematurely start re-optimizing the sub-set of S2L sub-LSPs. However, as mentioned in [RFC4875] Section 14.2, such sub-group based re-optimization procedure may result in data duplication that can be avoided if the entire P2MP-TE LSP tree is re-optimized using the Make-Before-Break method with a different LSP-ID, especially if the ingress node eventually receives PathErrs with the Notify error code for all S2L sub-LSPs of the P2MP-TE LSP tree.

In order to address above mentioned issues and to align re-optimization of P2MP-TE LSP with P2P LSP [RFC4736], there is a need for a mechanism to trigger re-optimization of the LSP tree by re-signaling all S2L sub-LSPs with a different LSP-ID. To meet this requirement, this document defines RSVP-TE signaling extensions for the ingress node to trigger the re-evaluation of the P2MP LSP tree on every hop that has a next-hop defined as a loose or abstract hop for one or more S2L sub-LSP path, and a mid-point LSR to signal to the ingress node that a preferable LSP tree exists (compared to the current path) or that the whole P2MP-TE LSP must be re-optimized (because of maintenance required on the TE LSP path) (see Section 4.1).

3.3. Existing Mechanism For Sub-Group-Based P2MP-TE LSP Re-optimization

Applying the procedures discussed in RFC4736 in conjunction with the Sub-Group-Based Re-Optimization procedures ([RFC4875], Section 14.2), an ingress node MAY trigger path re-evaluation requests for a set of S2L sub-LSPs in a single Path message using S2L sub-LSP descriptor list. Similarly, a mid-point LSR may send a PathErr with the Notify error code 25 and sub-code 6 containing a list of S2L sub-LSPs transiting through the LSR using an S2L sub-LSP descriptor list to notify the ingress node. This method can be used for re-optimizing a sub-group of S2L sub-LSPs within an LSP tree using the same LSP-ID.

This method can alleviate the scale issue associated with sending RSVP messages for individual S2L sub-LSPs. However, this procedure can lead to the following issues when used to re-optimize the LSP tree:

- Path message that is intended to carry the path re-evaluation request as defined in [RFC4736] with a full list of S2L sub-LSPs in S2L sub-LSPs descriptor list will be decomposed at branching LSRs, and only a subset of the S2L sub-LSPs that are routed over the same next-hop will be added in the descriptor list of the Path message propagated to downstream mid-point LSRs. Consequently, when a preferable path exists at such mid-point LSRs, the PathErr with the Notify error code can only include the sub-set of S2L sub-LSPs traversing the LSR. In this case, at the ingress node there is no way to distinguish which mode of re-optimization to invoke, i.e. sub-group based re-optimization using the same LSP-ID or tree based re-optimization using a different LSP-ID.

- An LSR may semantically fragment a large RSVP message (when a combined message may not be large enough to fit all S2L sub-LSPs). In this case, the ingress node may receive multiple PathErrs with sub-sets of S2L sub-LSPs in each (due to either the combined Path message getting fragmented or the combined PathErr message getting fragmented) and would require additional logic to determine how to re-optimize the LSP tree (for example, waiting for some time to aggregate all possible PathErr messages before taking an action). When fragmented, RSVP messages may arrive out of order, and the receiver has no way of knowing the beginning and end of the S2L sub-LSP list.

In order to address the above mentioned issues caused by RSVP message semantic fragmentation, this document defines new fragment identifier object for the S2L sub-LSP descriptor list when combining large number of S2L sub-LSPs in an RSVP message (see Section 4.2).

4. Signaling Extensions For Loosely Routed P2MP-TE LSP Re-optimization

4.1. Tree-Based Re-optimization

To evaluate a P2MP-TE LSP tree on mid-point LSRs that expand loose next-hop(s), an ingress node MAY send a Path message with "P2MP-TE Tree Re-evaluation Request (value TBA1)" defined in this document. The ingress node selects one of the S2L sub-LSPs of the P2MP-TE LSP tree transiting a mid-point LSR to trigger the re-evaluation request. The ingress node MAY send a re-evaluation request to each border LSR on the path of the LSP tree.

A mid-point LSR that expands loose next-hop(s) for one or more S2L sub-LSP path(s) does the following upon receiving a Path message with the "P2MP-TE Tree Re-evaluation Request" flag set:

- The mid-point LSR MUST check for a preferable P2MP-TE LSP tree by re-evaluating all S2L sub-LSP(s) that are expanded paths of the loose next-hops of the P2MP-TE LSP.

- If a preferable P2MP-TE LSP tree is found, the mid-point LSR MUST send an RSVP PathErr with the Notify error code 25 defined in [RFC3209] and sub-code "Preferable P2MP-TE Tree Exists (value TBA2)" defined in this document to the ingress node. The mid-point LSR, in turn, SHOULD NOT propagate the "P2MP-TE Tree Re-evaluation Request" flag in the subsequent RSVP Path messages sent downstream for the re-evaluated P2MP-TE LSP.

- If no preferable tree for P2MP-TE LSP can be found, the mid-point LSR that expands loose next-hop(s) for one or more S2L sub-LSP path(s) MUST propagate the request downstream by setting the "P2MP-TE Tree Re-evaluation Request" flag in the LSP_ATTRIBUTES Object of the RSVP Path message.

A mid-point LSR MAY send an unsolicited PathErr with the Notify error code and sub-code "Preferable P2MP-TE Tree Exists" to the ingress node to notify of a preferred P2MP-TE LSP tree when it determines it exists. In this case, the mid-point LSR that expands loose next-hop(s) for one or more S2L sub-LSP path(s) selects one of the S2L sub-LSP(s) of the P2MP-TE LSP tree to send this PathErr message to the ingress node. The mid-point LSR SHOULD consider how frequently it chooses to send such a PathErr - considering both that a PathErr may be lost on its transit to the ingress node and that the ingress node may choose not to re-optimize the LSP when such a PathErr is received.

The sending of an RSVP PathErr with the Notify error code and "Preferable P2MP-TE Tree Exists" sub-code to the ingress node notifies the ingress node of the existence of a preferable P2MP-TE LSP tree and upon receiving this PathErr, the ingress node SHOULD trigger re-optimization of the LSP using the MBB method with a different LSP-ID.

4.2. Sub-Group-Based Re-optimization Using Fragment Identifier

It might be preferable, as per [RFC4875], to re-optimize the entire P2MP-TE LSP by re-signaling all of its S2L sub-LSP(s) (Section 14.1, "Make-before-Break") or to re-optimize individual or group of S2L sub-LSP(s) i.e. individual or group of destination(s) (Section 14.2

"Sub-Group-Based Re-Optimization" in [RFC4875]), both using the same LSP-ID. For loosely routed S2L sub-LSPs, this can be achieved by using the procedures defined in [RFC4736] to re-optimize one or more S2L sub-LSP(s) of the P2MP-TE LSP.

An ingress node may trigger path re-evaluation requests using the procedures defined in [RFC4736] for a set of S2L sub-LSPs by combining multiple Path messages using an S2L sub-LSP descriptor list [RFC4875]. An S2L sub-LSP descriptor list is created using a series of S2L_SUB_LSP Objects as defined in [RFC4875]. Similarly, a mid-point LSR may send a PathErr with the Notify error code (value 25) and "Preferable Path Exists" (sub-code 6) containing a list of S2L sub-LSPs transiting through the LSR using an S2L sub-LSP descriptor list to notify the ingress node of preferable paths available.

As per [RFC4875] (Section 5.2.3, "Transit Fragmentation of Path State Information"), when a Path message is not large enough to fit all S2L sub-LSPs in the descriptor list, an LSR may semantically fragment the message. In this case, the LSR MUST add the S2L_SUB_LSP_FRAG Object defined in this document in the S2L sub-LSP descriptor to be able to rebuild the list from the received fragments that may arrive out of order.

The S2L_SUB_LSP_FRAG Object defined in this document is optional. However, a node MUST add the S2L_SUB_LSP_FRAG Object for each fragment in S2L sub-LSP descriptor when the RSVP message needs to be fragmented.

A mid-point LSR SHOULD wait to accumulate all S2L sub-LSPs before attempting to re-evaluate preferable path when a Path message for "Path Re-evaluation Request" is received with S2L_SUB_LSP_FRAG Object. If a mid-point LSR does not receive all fragments of the Path message (for example, when fragments are lost) within a configurable time interval, it SHOULD trigger re-evaluation of all S2L sub-LSPs of the P2MP-TE LSP transiting on the node. A mid-point LSR MUST receive at least one fragment of the Path message to trigger this behaviour.

An ingress node SHOULD wait to accumulate all S2L sub-LSPs before attempting to trigger re-optimization when a PathErr with Notify error code and "Preferable Path Exists" sub-code is received with a S2L_SUB_LSP_FRAG Object. If an ingress node does not receive all fragments of the PathErr message (for example, when fragments are lost) within a configurable time interval, it SHOULD trigger re-optimization of all S2L sub-LSPs of the P2MP-TE LSP transiting on the mid-point node that had sent the PathErr message. An ingress node MUST receive at least one fragment of the PathErr message to trigger this behaviour.

The S2L_SUB_LSP_FRAG Object defined in this document has a wider applicability in addition to the P2MP-TE LSP re-optimization. It can also be used (in Path and Resv messages) to setup a new P2MP-TE LSP, send other PathErr messages as well as Path Tear and Resv Tear messages for a set of S2L sub-LSPs. This is outside the scope of this document.

5. Message and Object Definitions

5.1. P2MP-TE Tree Re-evaluation Request Flag

In order to trigger a tree re-evaluation request, a new flag is defined in Attributes Flags TLV of the LSP_ATTRIBUTES Object [RFC5420] as follows:

Bit Number (TBA1, to be assigned by IANA): P2MP-TE Tree Re-evaluation Request flag

The "P2MP-TE Tree Re-evaluation Request" flag is meaningful in a Path message of a P2MP-TE S2L sub-LSP and is inserted by the ingress node using the message format defined in [RFC6510].

5.2. Preferable P2MP-TE Tree Exists Path Error Sub-code

In order to indicate to an ingress node that a preferable P2MP-TE LSP tree exists, the following new sub-code for PathErr with Notify error code 25 [RFC3209] is defined:

Sub-code (TBA2, to be assigned by IANA): Preferable P2MP-TE Tree Exists sub-code

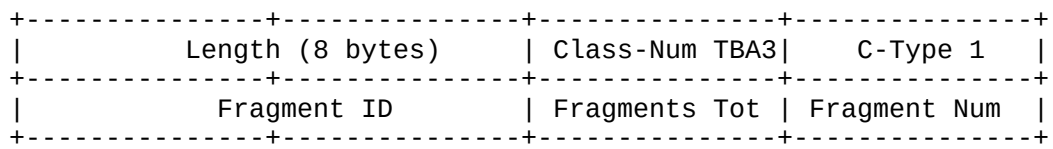
When a preferable path for P2MP-TE LSP tree exists, the mid-point LSR sends a solicited or unsolicited "Preferable P2MP-TE Tree Exists" sub-code with PathErr with Notify error code 25 to the ingress node of the P2MP-TE LSP.

5.3. Fragment Identifier For S2L sub-LSP Descriptor

The S2L_SUB_LSP Object [RFC4875] identifies a particular S2L sub-LSP belonging to the P2MP-TE LSP. An S2L sub-LSP descriptor list is created using a series of S2L_SUB_LSP Objects as defined in [RFC4875]. The RSVP message may need to be semantically fragmented [RFC4875] due to large number of S2L sub-LSPs added in the descriptor list, and such fragments may be received our of order. To be able to rebuild the fragmented S2L sub-LSP descriptor list correctly, the

following Object is defined to identify the fragments.

S2L_SUB_LSP_FRAG: Class-Num TBA3 by IANA



Fragment ID: 16-bit integer in the range of 1 to 65535.

This value is incremented for each new RSVP message that needs to be semantically fragmented. The fragment ID is reset to 1 when it reaches the maximum value of 65535. The scope of the fragment ID is limited to the RSVP message type (e.g. Path) carrying the fragment. In other words, fragment IDs do not have any correlation between different RSVP message types (e.g. Path and PathErr). The receiver does not check to ensure if the consecutive new RSVP messages (e.g. Path messages) are received with fragment IDs incremented by 1.

Fragments Total: 8-bit integer in the range of 1 to 255.

This value indicates the number of fragments sent for the given RSVP message. This value MUST be the same in all fragmented RSVP messages with a common Fragment ID.

Fragment Number: 8-bit integer in the range of 1 to 255.

This value indicates the position of this fragment in the given RSVP message.

The format of an S2L sub-LSP descriptor message is as follows:

```
<S2L sub-LSP descriptor> ::=
                  [ <S2L_SUB_LSP_FRAG> ]
                    <S2L_SUB_LSP>
                  [ <P2MP SECONDARY_EXPLICIT_ROUTE> ]
```

The S2L_SUB_LSP_FRAG Object is added before adding the S2L_SUB_LSP Object in the semantically fragmented RSVP message.

6. Compatibility

The LSP_ATTRIBUTES Object has been defined in [RFC5420] and its message formats in [RFC6510] with class numbers in the form 11bbbbbb, which ensures compatibility with non-supporting nodes. Per [RFC2205], nodes not supporting this extension will ignore the new flag defined for this Object in this document but forward it without modification.

The S2L_SUB_LSP_FRAG Object has been defined with class numbers in the form 11bbbbbb, which ensures compatibility with non-supporting nodes. Per [RFC2205], nodes not supporting this Object will ignore the Object but forward it without modification.

7. IANA Considerations

IANA is requested to administer assignment of new values for namespace defined in this document and summarized in this section.

7.1. P2MP-TE Tree Re-evaluation Request Flag

IANA maintains a name space for RSVP-TE TE parameters "Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Parameters" (see http://www.iana.org/assignments/rsvp-te-parameters). From the registries in this name space "Attribute Flags", allocation of new flag is requested (Section 5.1).

The following new flag is defined for the Attributes Flags TLV in the LSP_ATTRIBUTES Object [RFC5420]. The numeric value is to be assigned by IANA.

o P2MP-TE Tree Re-evaluation Request Flag:

Bit No	Attribute Flag Name	Carried in Path	Carried in Resv	Carried in RRO or ERO	Reference
TBA1 by IANA	P2MP-TE Tree Re-evaluation	Yes	No	No	This document

7.2. Preferable P2MP-TE Tree Exists Path Error Sub-code

IANA maintains a name space for RSVP protocol parameters "Resource Reservation Protocol (RSVP) Parameters" (see http://www.iana.org/assignments/rsvp-parameters). From the sub-registry "Sub-Codes - 25 Notify Error" in registry "Error Codes

and Globally-Defined Error Value Sub-Codes", allocation of a new error code is requested (Section 5.2).

As defined in [RFC3209], the Error Code 25 in the ERROR SPEC Object corresponds to PathErr with Notify error. This document adds a new sub-code for this PathErr as follows:

o Preferable P2MP-TE Tree Exists sub-code:

Sub-code value	Sub-code Description	PathErr Code	PathErr Name	Reference
TBA2 by IANA	Preferable P2MP-TE Tree Exists	25	Notify Error	This document

7.3. Fragment Identifier For S2L sub-LSP Descriptor

IANA maintains a name space for RSVP protocol parameters "Resource Reservation Protocol (RSVP) Parameters" (see http://www.iana.org/assignments/rsvp-parameters). From the registry "Class Names, Class Numbers, and Class Types", allocation of new Class-Num is requested (Section 5.3).

o S2L_SUB_LSP_FRAG Object:

Class-Num value	Description	Reference
TBA3 by IANA	S2L_SUB_LSP_FRAG	This document

8. Security Considerations

This document defines RSVP-TE signaling extensions to allow an ingress node of a P2MP-TE LSP to request the re-evaluation of the LSP tree downstream of a node, and for a mid-point LSR to notify the ingress node of the existence of a preferable tree by sending a PathErr. As per [RFC4736], in the case of a P2MP-TE LSP S2L sub-LSP spanning multiple domains, it may be desirable for a mid-point LSR to modify the RSVP PathErr message defined in this document to preserve confidentiality across domains.

This document also defines fragment identifier for the S2L sub-LSP descriptor when combining large number of S2L sub-LSPs in an RSVP

message and the message needs to be semantically fragmented. The introduction of the fragment identifier, by itself, introduces no additional information to signaling. For a general discussion on MPLS and GMPLS related security issues, see the MPLS/GMPLS security framework [RFC5920].

9. References

9.1. Normative References

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

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 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.

[RFC4736] Vasseur, JP., Ikejiri, Y. and Zhang, R, "Reoptimization of Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Loosely Routed Label Switched Path (LSP)", RFC 4736, November 2006.

[RFC4875] Aggarwal, R., Papadimitriou, D., and S. Yasukawa, "Extensions to Resource Reservation Protocol Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, May 2007.

[RFC5420] Farrel, A., Papadimitriou, D., Vasseur, JP., and Ayyangar, A., "Encoding of Attributes for MPLS LSP Establishment Using Resource Reservation Protocol Traffic Engineering (RSVP-TE)", RFC 5420, February 2009.

9.2. Informative References

[RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC5440] Vasseur, JP., Ed., and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, March 2009.

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

[RFC6510] Berger, L. and G. Swallow, "Resource Reservation Protocol (RSVP) Message Formats for Label Switched Path (LSP) Attributes Objects", RFC 6510, February 2012.

Acknowledgments

The authors would like to thank Loa Andersson, Sriganesh Kini, Curtis Villamizar, Dimitri Papadimitriou, Nobo Akiya, Vishnu Pavan Beeram and Joel M. Halpern for reviewing this document and providing many useful comments and suggestions. The authors would also like to thank Ling Zeng with Cisco Systems for implementing mechanisms defined in this document. A special thanks to Adrian Farrel for his thorough review of this document.

Author's Addresses

Tarek Saad (editor)
Cisco Systems

EMail: tsaad@cisco.com

Rakesh Gandhi (editor)
Cisco Systems

EMail: rgandhi@cisco.com

Zafar Ali
Cisco Systems

EMail: zali@cisco.com

Robert H. Venator
Defense Information Systems Agency

EMail: robert.h.venator.civ@mail.mil

Yuji Kamite
NTT Communications Corporation

EMail: y.kamite@ntt.com

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

H. Chen
Huawei Technologies
A. Liu
Ciena
T. Saad
Cisco Systems
F. Xu
Verizon
L. Huang
China Mobile

March 18, 2018

Extensions to RSVP-TE for LSP Egress Local Protection
draft-ietf-teas-rsvp-egress-protection-16.txt

Abstract

This document describes extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for locally protecting the egress node(s) of a Point-to-Point (P2P) or Point-to-Multipoint (P2MP) Traffic Engineered (TE) Label Switched Path (LSP).

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on September 19, 2018.

Table of Contents

1. Introduction

RFC 4090 describes two methods for locally protecting the transit nodes of a P2P LSP: one-to-one and facility protection. RFC 4875 specifies how these methods can be used to protect the transit nodes of a P2MP LSP. These documents do not discuss the procedures for locally protecting the egress node(s) of an LSP.

This document fills that void and specifies extensions to RSVP-TE for local protection of the egress node(s) of an LSP. Egress node and egress will be used exchangeably.

1.1. Egress Local Protection

In general, locally protecting an egress node of an LSP means that when the egress node fails, the traffic that the LSP carries will be delivered to its destination by the direct upstream node of the egress node to a backup egress node. Without protecting the egress node of the LSP, when the egress node fails, the traffic will be lost (i.e., the traffic will not be delivered to its destination).

Figure 1 shows an example of using backup LSPs to locally protect egress nodes L1 and L2 of a primary P2MP LSP starting from ingress R1. La and Lb are the designated backup egress nodes for primary egress nodes L1 and L2 respectively. The backup LSP for protecting L1 is from its upstream node R3 to backup egress node La and the backup LSP for protecting L2 is from R5 to Lb.

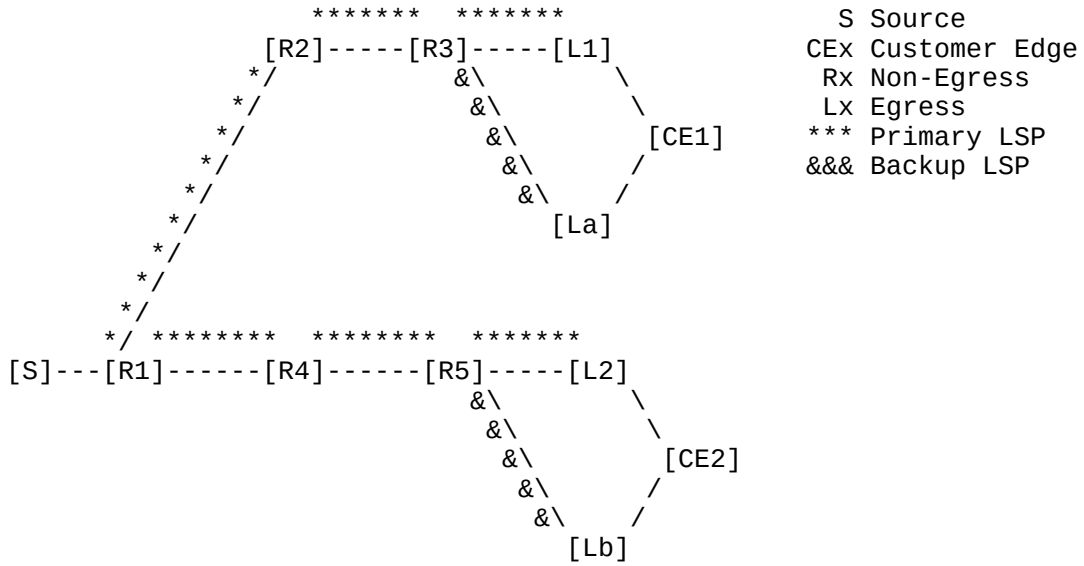


Figure 1: Backup LSP for Locally Protecting Egress

During normal operations, the traffic carried by the P2MP LSP is sent through R3 to L1, which delivers the traffic to its destination CE1. When R3 detects the failure of L1, R3 switches the traffic to the backup LSP to backup egress node La, which delivers the traffic to CE1. The time for switching the traffic is within tens of milliseconds.

The exact mechanism by which the failure of the primary egress node is detected by the upstream node R3 is out of the scope of this document.

In the beginning, the primary P2MP LSP from ingress R1 to primary egress nodes L1 and L2 is configured. It may be used to transport the traffic from source S connected to R1 to destinations CE1 and CE2 connected to L1 and L2 respectively.

To protect the primary egress nodes L1 and L2, one configures on the ingress R1 a backup egress node for L1, another backup egress node for L2 and other options. After the configuration, the ingress sends a Path message for the LSP with the information such as SEROs (refer to section 4.1) containing the backup egress nodes for protecting the primary egress nodes.

After receiving the Path message with the information, the upstream node of a primary egress node sets up a backup LSP to the corresponding backup egress node for protecting the primary egress node.

2. Conventions Used in This Document

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

3. Terminologies

The following terminologies are used in this document.

LSP: Label Switched Path

TE: Traffic Engineering

P2MP: Point-to-MultiPoint

P2P: Point-to-Point

LSR: Label Switching Router

RSVP: Resource ReSerVation Protocol

S2L: Source-to-Leaf

SERO: Secondary Explicit Route Object

RRO: Record Route Object

BFD: Bidirectional Forwarding Detection

VPN: Virtual Private Network

L3VPN: Layer 3 VPN

VRF: Virtual Routing and Forwarding

LFIB: Label Forwarding Information Base

UA: Upstream Assigned

PLR: Point of Local Repair

BGP: Border Gateway Protocol

CE: Customer Edge

PE: Provider Edge

4. Protocol Extensions

4.1. Extensions to SERO

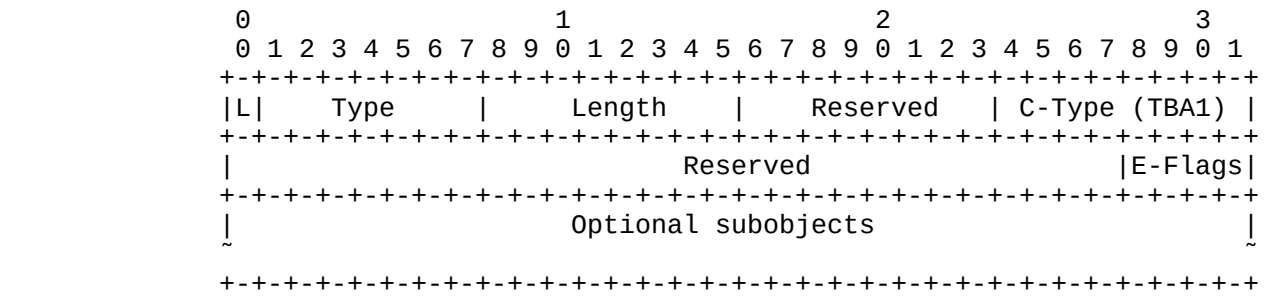
The Secondary Explicit Route object (SERO) is defined in RFC 4873. The format of the SERO is re-used.

The SERO used for protecting a primary egress node of a primary LSP may be added into the Path messages for the LSP and sent from the ingress node of the LSP to the upstream node of the egress node. It contains three subobjects.

The first subobject (refer to RFC 4873 Section 4.2) indicates the

branch node that is to originate the backup LSP (to a backup egress node). The branch node is typically the direct upstream node of the primary egress node of the primary LSP. If the direct upstream node does not support local protection against the failure of the primary egress node, the branch node can be any (upstream) node on the primary LSP. In this case, the backup LSP from the branch node to the backup egress node protects against failures on the segment of the primary LSP from the branch node to the primary egress node, including the primary egress node.

The second subobject is an egress protection subobject, which is a PROTECTION object with a new C-TYPE (TBA1). The format of the egress protection subobject is defined as follows:



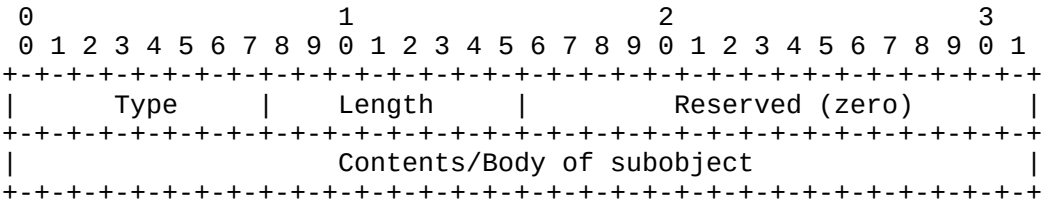
E-Flags are defined for egress local protection.

Bit 31 (Egress local protection flag): It is the least significant bit of the 32-bit word and is set to 1 indicating an egress local protection.

Bit 30 (S2L sub LSP backup desired flag): It is the second least significant bit of the 32-bit word and is set to 1 indicating S2L sub LSP (ref to RFC 4875) is desired for protecting an egress of a P2MP LSP.

The Reserved parts MUST be set to zero on transmission and MUST be ignored on receipt.

Four optional subobjects are defined. They are IPv4 and IPv6 primary egress node, IPv4 and IPv6 P2P LSP ID subobjects. IPv4 and IPv6 primary egress node subobjects indicate the IPv4 and IPv6 address of the primary egress node respectively. IPv4 and IPv6 P2P LSP ID subobjects contains the information for identifying IPv4 and IPv6 backup point-to-point (P2P) LSP tunnels respectively. Their contents are described in sections 4.1.1 through 4.1.2.2. They have the following format:



where Type is the type of a subobject, Length is the total size of the subobject in bytes, including Type, Length and Contents fields. The Reserved field MUST be set to zero on transmission and MUST be ignored on receipt.

The third (final) subobject (refer to RFC 4873 Section 4.2) in the SERO contains the egress node of the backup LSP, i.e., the address of the backup egress node in the place of the merge node.

After the upstream node of the primary egress node as the branch node receives the SERO and determines a backup egress node for the primary egress node, it computes a path from itself to the backup egress node and sets up a backup LSP along the path for protecting the primary egress node according to the information in the FAST_REROUTE object in the Path message. For example, if facility protection is desired, facility protection is provided for the primary egress node.

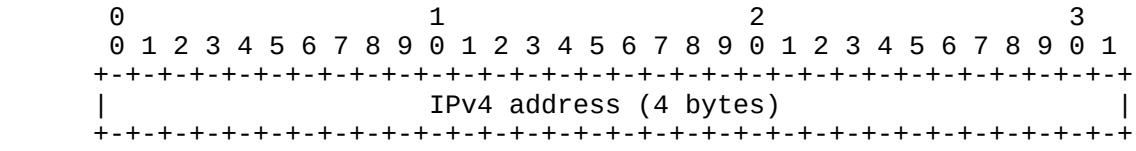
The upstream node constructs a new SERO based on the SERO received and adds the new SERO into the Path message for the backup LSP. The new SERO also contains three subobjects as the SERO for the primary LSP. The first subobject in the new SERO indicates the upstream node, which may be copied from the first subobject in the SERO received. The second subobject in the new SERO includes a primary egress node, which indicates the address of the primary egress node. The third one contains the backup egress node.

The upstream node updates the SERO in the Path message for the primary LSP. The egress protection subobject in the SERO contains a subobject called a P2P LSP ID subobject, which contains the information for identifying the backup LSP. The final subobject in the SERO indicates the address of the backup egress node.

4.1.1. Primary Egress Subobject

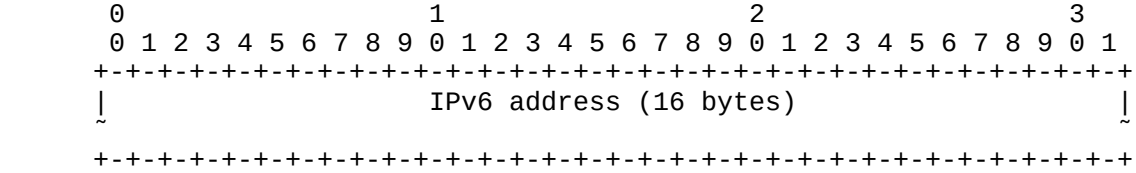
There are two primary egress subobjects. One is IPv4 primary egress subobject and the other is IPv6 primary egress subobject.

The Type of an IPv4 primary egress subobject is 1, and the body of the subobject is given below:



o IPv4 address: IPv4 address of the primary egress node

The Type of an IPv6 primary egress subobject is 2, and the body of the subobject is shown below:



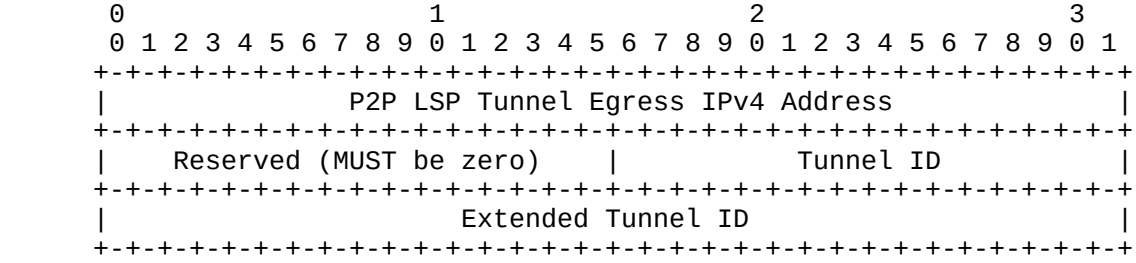
o IPv6 address: The IPv6 address of the primary egress node

4.1.2. P2P LSP ID Subobject

A P2P LSP ID subobject contains the information for identifying a backup point-to-point (P2P) LSP tunnel.

4.1.2.1. IPv4 P2P LSP ID Subobject

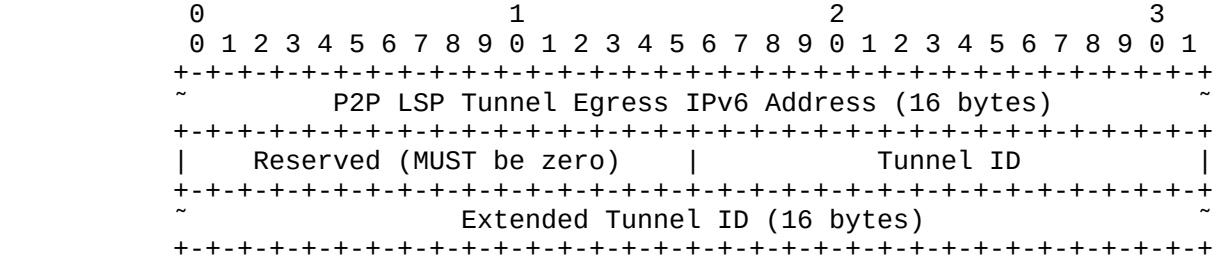
The Type of an IPv4 P2P LSP ID subobject is 3, and the body of the subobject is shown below:



o P2P LSP Tunnel Egress IPv4 Address:
  IPv4 address of the egress node of the tunnel
o Tunnel ID (ref to RFC 4875 and RFC 3209):
  A 16-bit identifier being constant over the life of the tunnel occupies the least significant 16 bits of the 32 bit word.
o Extended Tunnel ID (ref to RFC 4875 and RFC 3209):
  A 4-byte identifier being constant over the life of the tunnel

4.1.2.2. IPv6 P2P LSP ID Subobject

The Type of an IPv6 P2P LSP ID subobject is 4, and the body of the subobject is illustrated below:



- o P2P LSP Tunnel Egress IPv6 Address:
  IPv6 address of the egress node of the tunnel
- o Tunnel ID (ref to RFC 4875 and RFC 3209):
  A 16-bit identifier being constant over the life of the tunnel occupies the least significant 16 bits of the 32 bit word.
- o Extended Tunnel ID (ref to RFC 4875 and RFC 3209):
  A 16-byte identifier being constant over the life of the tunnel

5. Egress Protection Behaviors

5.1. Ingress Behavior

To protect a primary egress node of an LSP, the ingress MUST set the "label recording desired" flag and the "node protection desired" flag in the SESSION_ATTRIBUTE object.

If one-to-one backup or facility backup is desired to protect a primary egress node of an LSP, the ingress MUST include a FAST_REROUTE object and set the "One-to-One Backup Desired" or "Facility Backup Desired" flag respectively.

If S2L Sub LSP backup is desired to protect a primary egress node of a P2MP LSP, the ingress MUST set the "S2L Sub LSP Backup Desired" flag in an SERO object.

The decision to instantiate a backup egress for protecting the primary egress of an LSP can be initiated by either the ingress or the primary egress of that LSP, but not both.

A backup egress node MUST be configured on the ingress of an LSP to protect a primary egress node of the LSP if and only if the backup

egress node is not configured on the primary egress node (refer to section 5.2).

The ingress MUST send a Path message for the LSP with the objects above and the SEROs for protecting egress nodes of the LSP if protection of the egress nodes is desired. For each primary egress node of the LSP to be protected, the ingress MUST add an SERO object into the Path message if the backup egress node or some options are given. If the backup egress node is given, then the final subobject in the SERO contains it; otherwise the address in the final subobject is zero.

5.2. Primary Egress Behavior

To protect a primary egress node of an LSP, a backup egress node MUST be configured on the primary egress node of the LSP to protect the primary egress node if and only if the backup egress node is not configured on the ingress of the LSP (refer to section 5.1).

If the backup egress node is configured on the primary egress node of the LSP, the primary egress node MUST send its upstream node a Resv message for the LSP with an SERO for protecting the primary egress node. It sets the flags in the SERO in the same way as an ingress.

If the LSP carries the service traffic with a service label, the primary egress node sends its corresponding backup egress node the information about the service label as a UA label and the related forwarding.

5.3. Backup Egress Behavior

When a backup egress node receives a Path message for an LSP, it determines whether the LSP is used for egress local protection through checking the SERO with egress protection subobject in the message. If there is an egress protection subobject in the Path message for the LSP and the Egress local protection flag in the object is set to one, the LSP is the backup LSP for egress local protection. The primary egress node to be protected is in the primary egress subobject in the SERO.

When the backup egress node receives the information about a UA label and its related forwarding from the primary egress node, it uses the backup LSP label as a context label and creates a forwarding entry using the information about the UA label and the related forwarding. This forwarding entry is in a forwarding table for the primary egress node.

When the primary egress node fails, its upstream node switches the

traffic from the primary LSP to the backup LSP to the backup egress node, which delivers the traffic to its receiver such as CE using the backup LSP label as a context label to get the forwarding table for the primary egress node and the service label as UA label to find the forwarding entry in the table to forward the traffic to the receiver.

5.4. Transit Node and PLR Behavior

If a transit node of an LSP receives the Path message with the SEROs and it is not an upstream node of any primary egress node of the LSP as a branch node, it MUST forward them unchanged.

If the transit node is the upstream node of a primary egress node to be protected as a branch node, it determines the backup egress node, obtains a path for the backup LSP and sets up the backup LSP along the path. If the upstream node receives the Resv message with an SERO object, it MUST sends its upstream node the Resv message without the object.

The PLR (upstream node of the primary egress node as the branch node) MUST extract the backup egress node from the respective SERO object in either a Path or a Resv message. If no matching SERO object is found, the PLR tries to find the backup egress node, which is not the primary egress node but has the same IP address as the destination IP address of the LSP.

Note that if a backup egress node is not configured explicitly for protecting a primary egress node, the primary egress node and the backup egress node SHOULD have a same local address configured, and the cost to the local address on the backup egress node SHOULD be much bigger than the cost to the local address on the primary egress node. Thus primary egress node and backup egress node is considered as a virtual node. Note that the backup egress node is different from this local address (e.g., from the primary egress node' view). In other words, it is identified by an address different from this local address.

After obtaining the backup egress node, the PLR computes a backup path from itself to the backup egress node and sets up a backup LSP along the path. It excludes the segment including the primary egress node to be protected when computing the path. The PLR sends the primary egress node a Path message with an SERO for the primary LSP, which indicates the backup egress node by the final subobject in the SERO. The PLR puts an SERO into the Path messages for the backup LSP, which indicates the primary egress node.

The PLR MUST provide one-to-one backup protection for the primary egress node if the "One-to-One Backup Desired" flag is set in the

message; otherwise, it MUST provide facility backup protection if the "Facility Backup Desired flag" is set.

The PLR MUST set the protection flags in the RRO Sub-object for the primary egress node in the Resv message according to the status of the primary egress node and the backup LSP protecting the primary egress node. For example, it sets the "local protection available" and the "node protection" flag indicating that the primary egress node is protected when the backup LSP is up and ready for protecting the primary egress node.

5.4.1. Signaling for One-to-One Protection

The behavior of the upstream node of a primary egress node of an LSP as a PLR is the same as that of a PLR for one-to-one backup described in RFC 4090 except for that the upstream node as a PLR creates a backup LSP from itself to a backup egress node in a session different from the primary LSP.

If the LSP is a P2MP LSP and a primary egress node of the LSP is also a transit node (i.e., bud node), the upstream node of the primary egress node as a PLR creates a backup LSP from itself to each of the next hops of the primary egress node.

When the PLR detects the failure of the primary egress node, it switches the packets from the primary LSP to the backup LSP to the backup egress node. For the failure of the bud node of a P2MP LSP, the PLR also switches the packets to the backup LSPs to the bud node’s next hops, where the packets are merged into the primary LSP.

5.4.2. Signaling for Facility Protection

Except for backup LSP and downstream label, the behavior of the upstream node of the primary egress node of a primary LSP as a PLR follows the PLR behavior for facility backup described in RFC 4090.

For a number of primary P2P LSPs going through the same PLR to the same primary egress node, the primary egress node of these LSPs MAY be protected by one backup LSP from the PLR to the backup egress node designated for protecting the primary egress node.

The PLR selects or creates a backup LSP from itself to the backup egress node. If there is a backup LSP that satisfies the constraints given in the Path message, then this one is selected; otherwise, a new backup LSP to the backup egress node is created.

After getting the backup LSP, the PLR associates the backup LSP with a primary LSP for protecting its primary egress node. The PLR

records that the backup LSP is used to protect the primary LSP against its primary egress node failure and MUST include an SERO object in the Path message for the primary LSP. The object MUST contain the backup LSP ID. It indicates that the primary egress node MUST send the backup egress node the service label as UA label and the information about forwarding the traffic to its destination using the label if there is a service carried by the LSP and the primary LSP label as UA label if the label is not implicit null. How UA label is sent is out of scope for this document.

When the PLR detects the failure of the primary egress node, it redirects the packets from the primary LSP into the backup LSP to backup egress node and keeps the primary LSP label from the primary egress node in the label stack if the label is not implicit null. The backup egress node delivers the packets to the same destinations as the primary egress node using the backup LSP label as context label and the labels under as UA labels.

5.4.3. Signaling for S2L Sub LSP Protection

The S2L Sub LSP Protection uses a S2L Sub LSP (ref to RFC 4875) as a backup LSP to protect a primary egress node of a P2MP LSP. The PLR MUST determine to protect a primary egress node of a P2MP LSP via S2L sub LSP protection when it receives a Path message with flag "S2L Sub LSP Backup Desired" set.

The PLR MUST set up the backup S2L sub LSP to the backup egress node, create and maintain its state in the same way as of setting up a source to leaf (S2L) sub LSP defined in RFC 4875 from the signaling's point of view. It computes a path for the backup LSP from itself to the backup egress node, constructs and sends a Path message along the path, receives and processes a Resv message responding to the Path message.

After receiving the Resv message for the backup LSP, the PLR creates a forwarding entry with an inactive state or flag called inactive forwarding entry. This inactive forwarding entry is not used to forward any data traffic during normal operations.

When the PLR detects the failure of the primary egress node, it changes the forwarding entry for the backup LSP to active. Thus, the PLR forwards the traffic to the backup egress through the backup LSP, which sends the traffic to its destination.

5.4.4. PLR Procedures during Local Repair

When the upstream node of a primary egress node of an LSP as a PLR detects the failure of the primary egress node, it follows the

procedures defined in section 6.5 of RFC 4090. It SHOULD notify the ingress about the failure of the primary egress node in the same way as a PLR notifies the ingress about the failure of a transit node.

Moreover, the PLR MUST let the upstream part of the primary LSP stay alive after the primary egress node fails through sending Resv message to its upstream node along the primary LSP. The downstream part of the primary LSP from the PLR to the primary egress node SHOULD be removed. When a bypass LSP from the PLR to a backup egress node protects the primary egress node, the PLR MUST NOT send any Path message for the primary LSP through the bypass LSP to the backup egress node.

In the local revertive mode, the PLR will re-signal each of the primary LSPs that were routed over the restored resource once it detects that the resource is restored. Every primary LSP successfully re-signaled along the restored resource will be switched back.

Note that the procedure for protecting the primary egress node is triggered on the PLR if the primary egress node failure is determined. If link (from PLR to primary egress node) failure and primary egress node alive are determined, then link protection procedure is triggered on the PLR. How to determine these is out of scope for this document.

6. Application Traffic Considerations

This section focuses on an example with application traffic carried by P2P LSPs.

6.1. A Typical Application

L3VPN is a typical application. Figure 2 below shows a simple VPN, which consists of two CEs, CE1 and CE2, connected to two PEs, R1 and L1, respectively. There is a P2P LSP from R1 to L1, which is represented by stars (****). This LSP is called the primary LSP. R1 is the ingress of the LSP and L1 is the (primary) egress node of the LSP. R1 sends the VPN traffic received from CE1 through the P2P LSP to L1, which delivers the traffic to CE2. R1 sends the VPN traffic with a LSP label and a VPN label via the LSP. When the traffic reaches the egress node L1 of the LSP, L1 pops the LSP label and uses the VPN label to deliver the traffic to CE2.

In previous solutions based on ingress protection to protect the VPN traffic against failure of the egress node L1 of the LSP, when the egress node fails, the ingress R1 of the LSP does the reroute (refer

to Figure 2). This solution entailed:

1. A multi-hop BFD session between ingress R1 and egress node L1 of primary LSP. The BFD session is represented by dots (....).

2. A backup LSP from ingress R1 to backup egress node La, which is indicated by ands (&&&&).

3. La sends R1 a VPN backup label and related information via BGP.

4. R1 has a VRF with two sets of routes for CE2: one set uses the primary LSP and L1 as next hop; the other uses the backup LSP and La as next hop.

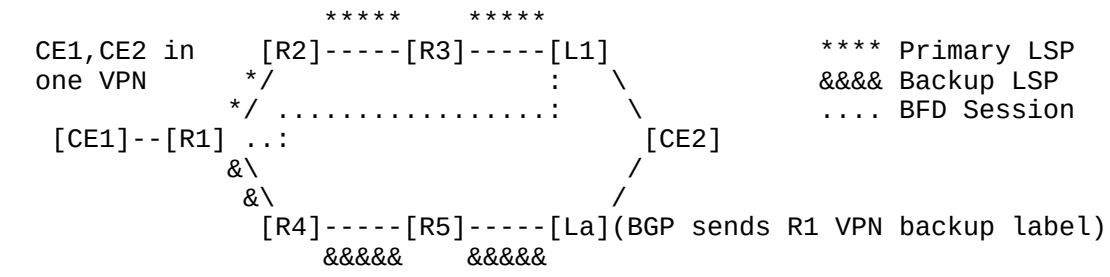


Figure 2: Protect Egress for L3VPN Traffic

In normal operations, R1 sends the VPN traffic from CE1 through the primary LSP with the VPN label received from L1 as inner label to L1, which delivers the traffic to CE2 using the VPN label.

When R1 detects the failure of L1, R1 sends the traffic from CE1 via the backup LSP with the VPN backup label received from La as inner label to La, which delivers the traffic to CE2 using the VPN backup label.

The solution defined in this document using egress local protection for protecting L3VPN traffic entails (refer to Figure 3):

1. A BFD session between R3 (i.e., upstream of L1) and egress node L1 of the primary LSP. This is different from the BFD session in Figure 2, which is multi-hop between ingress R1 and egress node L1. The PLR R3 is closer to L1 than the ingress R1. It may detect the failure of the egress node L1 faster and more reliably. Therefore, this solution can provide faster protection for failure of an egress node.

2. A backup LSP from R3 to backup egress node La. This is different from the backup LSP in Figure 2, which is an end to end LSP from ingress R1 to backup egress node La.

3. Primary egress node L1 sends backup egress node La the VPN label as UA label and related information. The backup egress node La uses the backup LSP label as a context label and creates a forwarding entry using the VPN label in a LFIB for the primary egress node L1.

4. L1 and La is virtualized as one node (or address). R1 has a VRF with one set of routes for CE2, using the primary LSP from R1 to L1 and virtualized node as next hop. This can be achieved by configuring a same local address on L1 and La, using the address as a destination of the LSP and BGP next hop for the VPN traffic. The cost to L1 is configured to be less than the cost to La.

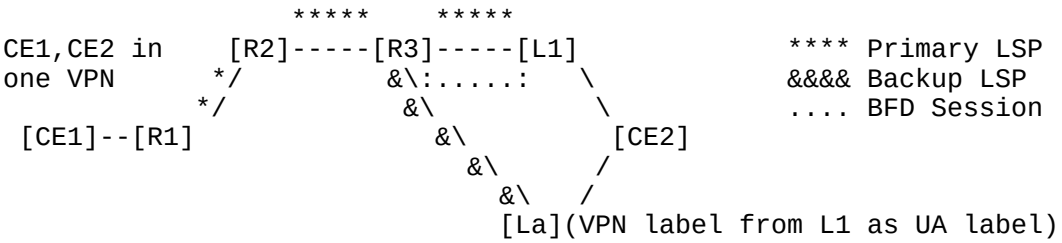


Figure 3: Locally Protect Egress for L3VPN Traffic

In normal operations, R1 sends the VPN traffic from CE1 via the primary LSP with the VPN label as inner label to L1, which delivers the traffic to CE2 using the VPN label.

When the primary egress node L1 fails, its upstream node R3 detects it and switches the VPN traffic from the primary LSP to the backup LSP to La, which delivers the traffic to CE2 using the backup LSP label as a context label to get the LFIB for L1 and the VPN label as UA label to find the forwarding entry in the LFIB to forward the traffic to CE2.

6.2. PLR Procedure for Applications

When the PLR gets a backup LSP from itself to a backup egress node for protecting a primary egress node of a primary LSP, it includes an SERO object in the Path message for the primary LSP. The object contains the ID information of the backup LSP and indicates that the primary egress node sends the backup egress node the application traffic label (e.g., the VPN label) as UA label when needed.

6.3. Egress Procedures for Applications

When a primary egress node of an LSP sends the ingress of the LSP a label for an application such as a VPN label, it sends the backup egress node for protecting the primary egress node the label as a UA label. Exactly how the label is sent is out of scope for this document.

When the backup egress node receives a UA label from the primary egress node, it adds a forwarding entry with the label into the LFIB for the primary egress node. When the backup egress node receives a packet from the backup LSP, it uses the top label as a context label to find the LFIB for the primary egress node and the inner label to deliver the packet to the same destination as the primary egress node according to the LFIB.

7. Security Considerations

This document builds upon existing work, specifically the security considerations of RFCs 4090, 4875, 3209 and 2205 continue to apply. Additionally, protecting a primary egress node of a P2P LSP carrying service traffic through a backup egress node requires an out-of-band communication between the primary egress node and the backup egress node, in order for the primary egress node to convey a service label as UA label and its related forwarding information to the backup egress node. It is important to confirm that the identifiers used to identify the primary and backup egress nodes in the LSP are verified to match with the identifiers used in the out-of-band protocol (such as BGP).

8. IANA Considerations

IANA maintains a registry called "Class Names, Class Numbers, and Class Types" under "Resource Reservation Protocol (RSVP) Parameters". IANA is requested to assign a new C-Type under PROTECTION object class, Class Number 37:

Value	Description	Definition
TBA1(suggested value 3)	Egress Protection	Section 4.1

IANA is requested to create and maintain a new registry under PROTECTION object class (Class Number 37) and Egress Protection (C-Type TBA1). Initial values for the registry are given below. The future assignments are to be made through IETF Review (RFC 8216).

Value	Name	Definition
0	Reserved	
1	IPv4_PRIMARY_EGRESS	Section 4.1.1
2	IPv6_PRIMARY_EGRESS	Section 4.1.1
3	IPv4_P2P_LSP_ID	Section 4.1.2
4	IPv6_P2P_LSP_ID	Section 4.1.2
5-127	Unassigned	
128-255	Reserved	

9. Co-authors and Contributors

1. Co-authors

Ning So
Tata
E-mail: ningso01@gmail.com

Mehmet Toy
Verizon
E-mail: mehmet.toy@verizon.com

Lei Liu
Fujitsu
E-mail: lliu@us.fujitsu.com

Zhenbin Li
Huawei Technologies
Email: lizhenbin@huawei.com

2. Contributors

Boris Zhang
Telus Communications
Email: Boris.Zhang@telus.com

Nan Meng
Huawei Technologies
Email: mengnan@huawei.com

Prejeeth Kaladharan
Huawei Technologies
Email: prejeeth@gmail.com

Vic Liu
China Mobile
Email: liu.cmri@gmail.com

10. Acknowledgement

The authors would like to thank Richard Li, Nobo Akiya, Lou Berger, Jeffrey Zhang, Lizhong Jin, Ravi Torvi, Eric Gray, Olufemi Komolafe, Michael Yue, Daniel King, Rob Rennison, Neil Harrison, Kannan Sampath, Yimin Shen, Ronhazli Adam and Quintin Zhao for their valuable comments and suggestions on this draft.

11. References

11.1. Normative References

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <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>.

[RFC4873] Berger, L., Bryskin, I., Papadimitriou, D., and A. Farrel, "GMPLS Segment Recovery", RFC 4873, DOI 10.17487/RFC4873, May 2007, <https://www.rfc-editor.org/info/rfc4873>.

[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>.

11.2. Informative References

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <https://www.rfc-editor.org/info/rfc2205>.

[RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <https://www.rfc-editor.org/info/rfc5331>.

[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>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <https://www.rfc-editor.org/info/rfc3473>.

[FRAMEWK] Shen, Y., Jeyananth, M., Decraene, B., and H. Gredler, "MPLS Egress Protection Framework", draft-shen-mpls-egress-protection-framework , October 2016.

Authors' Addresses

Huaimo Chen
Huawei Technologies
Boston, MA
USA

Email: huaimo.chen@huawei.com

Autumn Liu
Ciena
USA

Email: hliu@ciena.com

Tarek Saad
Cisco Systems

Email: tsaad@cisco.com

Fengman Xu
Verizon
2400 N. Glenville Dr
Richardson, TX 75082
USA

Email: fengman.xu@verizon.com

Lu Huang
China Mobile
No.32 Xuanwumen West Street, Xicheng District
Beijing, 100053
China

Email: huanglu@chinamobile.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

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on September 19, 2018.

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.

- 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.

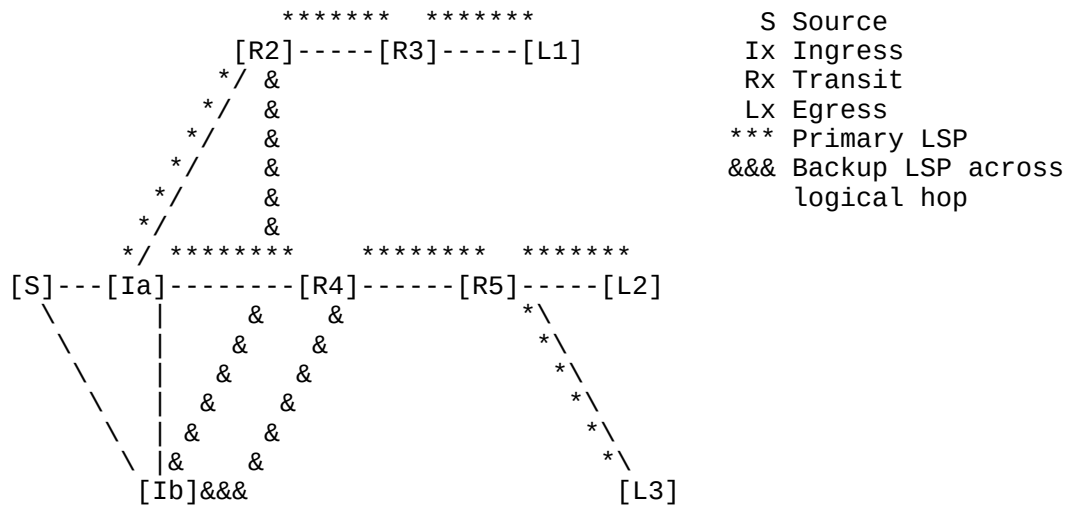


Figure 1: Ingress Local Protection

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:

- Setting up the necessary backup LSP forwarding state based on the information received for ingress local protection;
- Detecting the primary ingress failure and providing the fast repair (as discussed in Sections 2 and 3);
- Maintaining the RSVP-TE control plane state until a global repair is done; and
- 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:

- Backup ingress IP address indicating the backup ingress,
- 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,
- Label and Routes indicating the first hops of the primary LSP, each of which is paired with its label, and
- 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

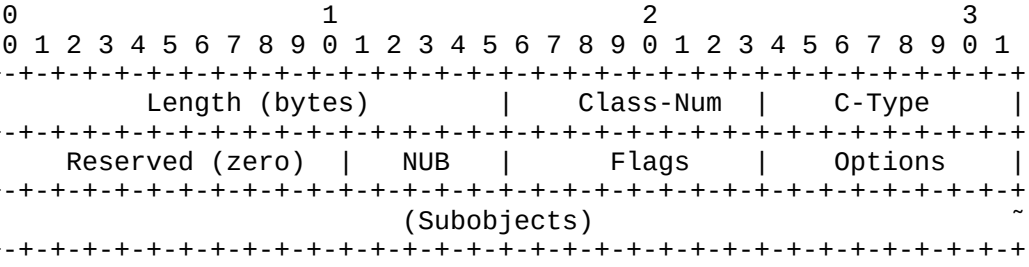
4.1.1. Class Number and Class Type

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:



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.

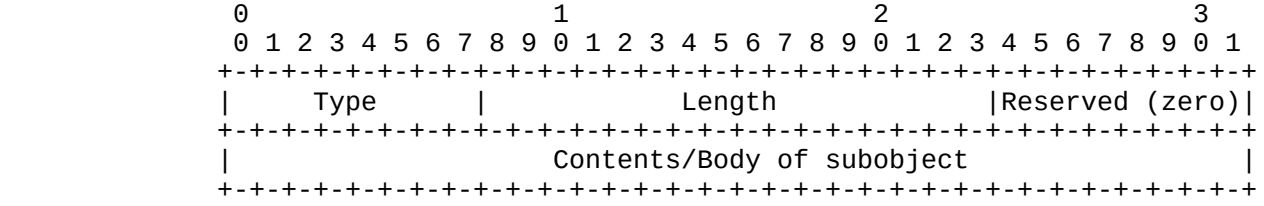
- o Bandwidth protection: The backup ingress MUST set this flag if the backup LSPs guarantee to provide desired bandwidth for the protected LSP against the primary ingress failure.

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:

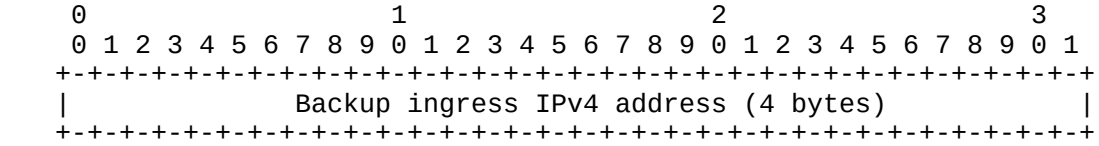


where Type is the type of a subobject, Length is the total size of the subobject in bytes, including Type, Length and Contents fields.

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

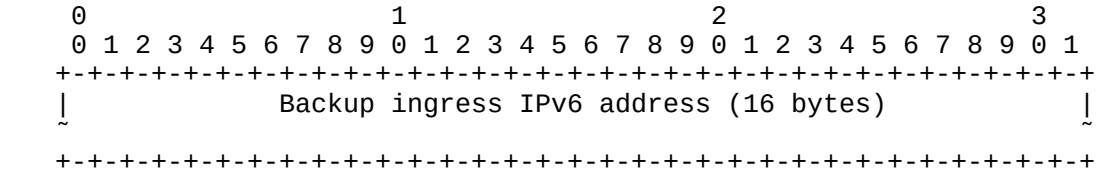
have a Backup Ingress IPv4 Address subobject containing an IPv4 address belonging to the backup ingress if IPv4 is used. The Type of the subobject is 1, and the body of the subobject is given below:



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

4.1.4. Subobject: Backup Ingress IPv6 Address

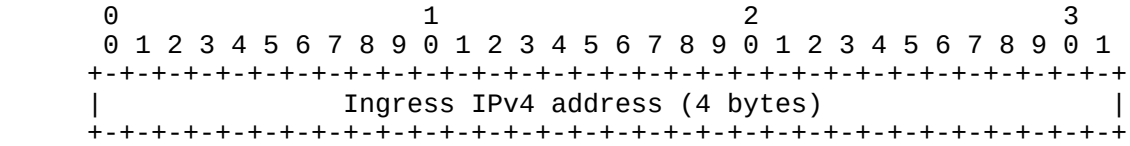
When the primary ingress of a protected LSP sends a PATH message with an INGRESS_PROTECTION object to the backup ingress, the object MUST have a Backup Ingress IPv6 Address subobject containing an IPv6 address belonging to the backup ingress if IPv6 is used. The Type of the subobject is 2, the body of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



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

4.1.5. Subobject: Ingress IPv4 Address

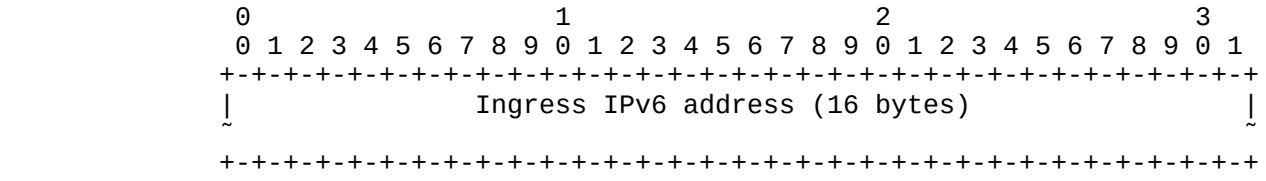
The INGRESS_PROTECTION object may have an Ingress IPv4 Address subobject containing an IPv4 address belonging to the primary ingress if IPv4 is used. The Type of the subobject is 3. The subobject has the following body:



Ingress IPv4 address: An IPv4 host address of ingress

4.1.6. Subobject: Ingress IPv6 Address

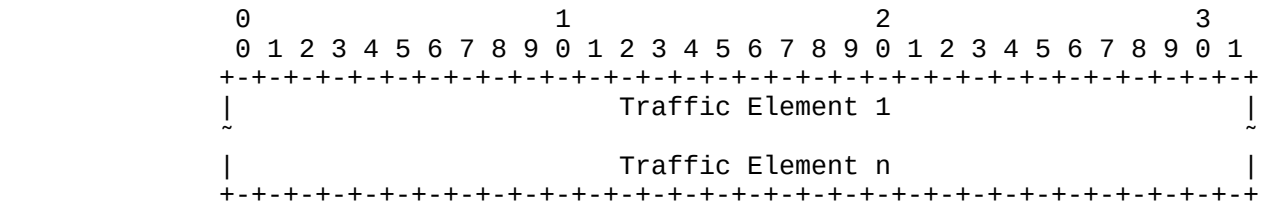
The INGRESS_PROTECTION object may have an Ingress IPv6 Address subobject containing an IPv6 address belonging to the primary ingress if IPv6 is used. The Type of the subobject is 4. The subobject has the following body:



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:



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

- 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.

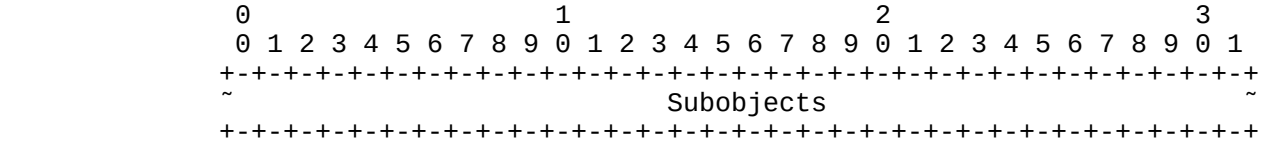
- IPv4 Prefix Traffic: Each of the Traffic Elements is an IPv4 prefix, containing an 8-bit prefix length followed by an IPv4 address prefix, whose length, in bits, is specified by the prefix length, padded to a byte boundary.

- 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:



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.

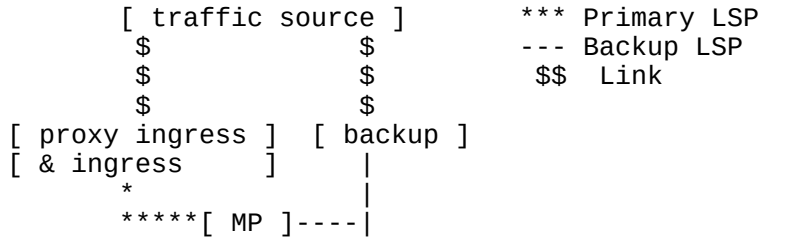


Figure 2: Example Protected LSP with Proxy Ingress Node

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.

- 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.

- 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].

- 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.

- 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
Expires February 24, 2017

F. Zhang, Ed.
Huawei
O. Gonzalez de Dios, Ed.
Telefonica Global CTO
M. Hartley
Z. Ali
Cisco
C. Margaria

August 24, 2016

RSVP-TE Extensions for Collecting SRLG Information
draft-ietf-teas-rsvp-te-srlg-collect-08

Abstract

This document provides extensions for the Resource ReserVation Protocol-Traffic Engineering (RSVP-TE), including GMPLS, to support automatic collection of Shared Risk Link Group (SRLG) information for the TE link formed by a Label Switched Path (LSP).

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 February 24, 2017.

Table of Contents

1. Introduction

It is important to understand which Traffic Engineering (TE) links in the network might be at risk from the same failures. In this sense, a set of links can constitute a 'shared risk link group' (SRLG) if they share a resource whose failure can affect all links in the set [RFC4202].

On the other hand, as described in [RFC4206] and [RFC6107], H-LSP (Hierarchical LSP) or S-LSP (stitched LSP) can be used for carrying one or more other LSPs. Both of the H-LSP and S-LSP can be formed as a TE link. In such cases, it is important to know the SRLG information of the LSPs that will be used to carry further LSPs.

This document provides a signaling mechanism to collect the SRLGs used by a LSP, which can then be advertized as properties of the TE-link formed by that LSP.

1.1. Applicability Example: Dual Homing

An interesting use case for the SRLG collection procedures defined in this document is achieving LSP diversity in a dual homing scenario. The use case is illustrated in Figure 1, when the overlay model is applied as defined in RFC 4208 [RFC4208] . In this example, the exchange of routing information over the User-Network Interface (UNI) is prohibited by operator policy.

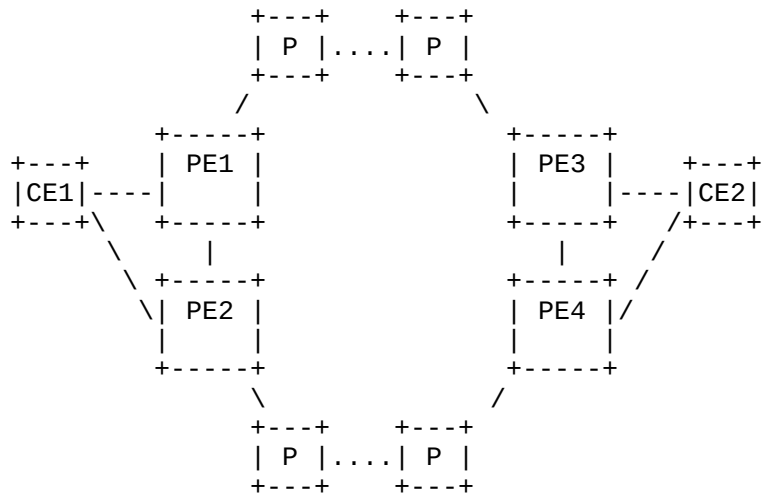


Figure 1: Dual Homing Configuration

Single-homed customer edge (CE) devices are connected to a single provider edge (PE) device via a single UNI link (which could be a bundle of parallel links, typically using the same fiber cable). This single UNI link can constitute a single point of failure. Such a single point of failure can be avoided if the CE device is connected to two PE devices via two UNI interfaces as depicted in Figure 1 above for CE1 and CE2, respectively.

For the dual-homing case, it is possible to establish two connections

(LSPs) from the source CE device to the same destination CE device where one connection is using one UNI link to PE1, for example, and the other connection is using the UNI link to PE2. In order to avoid single points of failure within the provider network, it is necessary to also ensure path (LSP) diversity within the provider network in order to achieve end-to-end diversity for the two LSPs between the two CE devices CE1 and CE2. This use case describes how it is possible to achieve path diversity within the provider network based on collected SRLG information. As the two connections (LSPs) enter the provider network at different PE devices, the PE device that receives the connection request for the second connection needs to know the additional path computation constraints such that the path of the second LSP is disjoint with respect to the already established first connection.

As SRLG information is normally not shared between the provider network and the client network, i.e., between PE and CE devices, the challenge is how to solve the diversity problem when a CE is dual-homed. The RSVP extensions for collecting SRLG information defined in this document make it possible to retrieve SRLG information for an LSP and hence solve the dual-homing LSP diversity problem. For example, CE1 in Figure 1 may have requested an LSP1 to CE2 via PE1 that is routed via PE3 to CE2. CE1 can then subsequently request an LSP2 to CE2 via PE2 with the constraint that it needs to be maximally SRLG disjoint with respect to LSP1. PE2, however, does not have any SRLG information associated with LSP1, which is needed as input for its constraint-based path computation function. If CE1 is capable of retrieving the SRLG information associated with LSP1 from PE1, it can pass this discovered information to PE2 as part of the LSP2 setup request (RSVP PATH message) in an EXCLUDE_ROUTE Object (XRO) or Explicit Exclusion Route Subobject (EXRS) as described in [RFC4874], and PE2 can now calculate a path for LSP2 that is SRLG disjoint with respect to LSP1. The SRLG information associated with LSP1 can be retrieved when LSP1 is established or at any time before LSP2 is setup.

When CE1 sends the setup request for LSP2 to PE2, it can also request the collection of SRLG information for LSP2 and send that information to PE1 by re-signaling LSP1 with SRLG-exclusion based on LSP2’s discovered SRLGs. This will ensure that the two paths for the two LSPs remain mutually diverse, which is important when the provider network is capable of restoring connections that failed due to a network failure (fiber cut) in the provider network.

Note that the knowledge of SRLG information even for multiple LSPs does not allow a CE device to derive the provider network topology based on the collected SRLG information. It would, however, be possible for an entity controlling multiple CE devices to derive some

information related to the topology. This document therefore allows PE devices to control the communication of SRLGs outside the provider network if desired.

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. RSVP-TE Requirements

The SRLG-collection process takes place in three stages:

- The LSP’s ingress node requests that SRLG collection take place;
- SRLG data is added to the Path and Resv ROUTE_RECORD Objects (RROs) by all nodes during signaling;
- Changes to previously-signaled SRLG data are made by sending updated Path and Resv messages as required.

3.1. SRLG Collection Indication

The ingress node of the LSP needs be capable of indicating whether the SRLG information of the LSP is to be collected during the signaling procedure of setting up an LSP. There is no need for SRLG information to be collected without an explicit request for it being made by the ingress node.

It may be preferable for the SRLG collection request to be understood by all nodes along the LSP’s path, or it may be more important for the LSP to be established successfully even if it traverses nodes that cannot supply SRLG information or have not implemented the procedures specified in this document. It is desirable for the ingress node to make the SRLG collection request in a manner that best suits its own policy.

3.2. SRLG Collection

If requested, the SRLG information is collected during the setup of an LSP. SRLG information is added by each hop to the Path RRO during Path message processing. The same information is also added to the Resv RRO during Resv processing at each hop.

3.3. SRLG Update

When the SRLG information of an existing LSP for which SRLG information was collected during signaling changes, the relevant nodes of the LSP need to be capable of updating the SRLG information of the LSP. This means that the signaling procedure needs to be capable of updating the new SRLG information.

3.4. SRLG ID definition

The identifier of an SRLG (SRLG ID) is defined as a 32-bit quantity in [RFC4202]. This definition is used in this document.

4. Encodings

4.1. SRLG Collection Flag

In order to indicate to nodes that SRLG collection is desired, this document defines a new flag in the Attribute Flags TLV (see RFC 5420 [RFC5420]). This document defines a new SRLG collection flag in the Attribute Flags TLV (see RFC 5420 [RFC5420]). A node that wishes to indicate that SRLG collection is desired MUST set this flag in an Attribute Flags TLV in an LSP_REQUIRED_ATTRIBUTES Object if collection is to be mandatory, or an LSP_ATTRIBUTES Object if collection is desired but not mandatory.

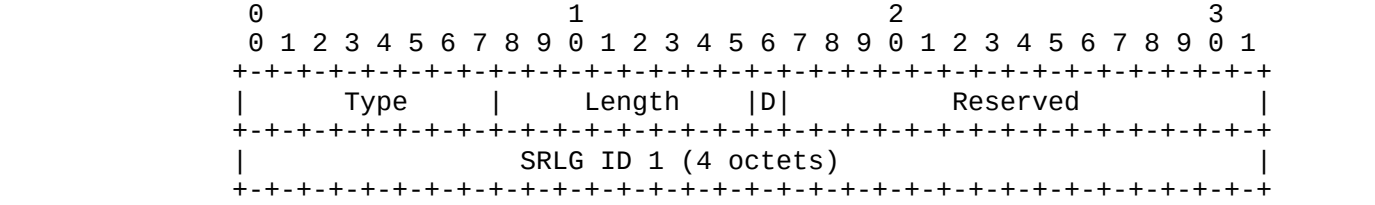
o Bit Number (specified in Section 8.1): SRLG Collection flag

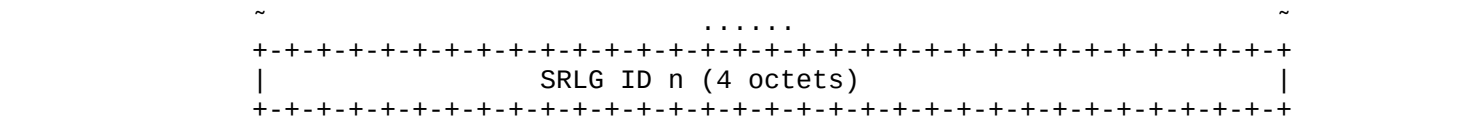
The SRLG Collection flag is meaningful on a Path message. If the SRLG Collection flag is set to 1, it means that the SRLG information SHOULD be reported to the ingress and egress node along the setup of the LSP.

The rules for the processing of the Attribute Flags TLV are not changed.

4.2. RRO SRLG sub-object

This document defines a new RRO sub-object (ROUTE_RECORD sub-object) to record the SRLG information of the LSP. Its format is modeled on the RRO sub-objects defined in RFC 3209 [RFC3209].





Type (8 bits)

The type of the sub-object. The value is specified in Section 8.2.

Length (8 bits)

The Length field contains the total length of the sub-object in octets, including the Type and Length fields. The Length depends on the number of SRLG IDs.

Direction bit (D-bit) (1 bit)

If not set, the SRLGs contained in this sub-object apply to the downstream direction. If set, they apply to the upstream direction.

Reserved (15 bits)

This 15-bit field is reserved. It SHOULD be set to zero on transmission and MUST be ignored on receipt.

SRLG ID (4 octets)

This field contains one SRLG ID. There is one SRLG ID field per SRLG collected. There MAY be multiple SRLG ID fields in an SRLG sub-object.

A node MUST NOT push a SRLG sub-object in the RECORD_ROUTE without also pushing either a IPv4 sub-object, a IPv6 sub-object, a Unnumbered Interface ID sub-object or a Path Key sub-object.

As described in RFC 3209 [RFC3209], the RECORD_ROUTE object is managed as a stack. The SRLG sub-object MUST be pushed by the node before the node IP address or link identifier. The SRLG-sub-object SHOULD be pushed after the Attribute sub-object, if present, and after the LABEL sub-object, if requested. It MUST be pushed within the hop to which it applies.

RFC 5553 [RFC5553] describes mechanisms to carry a PKS (Path Key Sub-object) in the RRO so as to facilitate confidentiality in the signaling of inter-domain TE LSPs, and allows the path segment that needs to be hidden (that is, a Confidential Path Segment (CPS)) to be replaced in the RRO with a PKS. If the CPS contains SRLG Sub-objects, these MAY be retained in the RRO by adding them again after

the PKS Sub-object in the RRO. The CPS is defined in RFC 5520 [RFC5520].

The rules for the processing of the LSP_REQUIRED_ATTRIBUTES, LSP_ATTRIBUTE and ROUTE_RECORD Objects are not changed.

5. Signaling Procedures

The ingress node of the LSP MUST be capable of indicating whether the SRLG information of the LSP is to be collected during the signaling procedure of setting up an LSP.

5.1. SRLG Collection

Per RFC 3209 [RFC3209], an ingress node initiates the recording of the route information of an LSP by adding a RRO to a Path message. If an ingress node also desires SRLG recording, it MUST set the SRLG Collection Flag in the Attribute Flags TLV which MAY be carried either in an LSP_REQUIRED_ATTRIBUTES Object when the collection is mandatory, or in an LSP_ATTRIBUTES Object when the collection is desired, but not mandatory.

A node MUST NOT add SRLG information without an explicit request for it being made by the ingress node in the Path message.

When a node receives a Path message which carries an LSP_REQUIRED_ATTRIBUTES Object with the SRLG Collection Flag set, if local policy determines that the SRLG information is not to be provided to the endpoints, it MUST return a PathErr message with:

- Error Code 2 (policy) and
- Error subcode "SRLG Recording Rejected" (see Section 8.3 for value)

to reject the Path message.

When a node receives a Path message which carries an LSP_ATTRIBUTES Object with the SRLG Collection Flag set, if local policy determines that the SRLG information is not to be provided to the endpoints, the Path message MUST NOT be rejected due to the SRLG recording restriction and the Path message MUST be forwarded without any SRLG sub-object(s) added to the RRO of the corresponding outgoing Path message.

If local policy permits the recording of the SRLG information, the processing node SHOULD add local SRLG information, as defined below, to the RRO of the corresponding outgoing Path message. The processing node MAY add multiple SRLG sub-objects to the RRO if necessary. It then forwards the Path message to the next node in the downstream direction. The processing node MUST retain a record of the

SRLG recording request for reference during Resv processing described below.

If the addition of SRLG information to the RRO would result in the RRO exceeding its maximum possible size or becoming too large for the Path message to contain it, the requested SRLGs MUST NOT be added. If the SRLG collection request was contained in an LSP_REQUIRED_ATTRIBUTES Object, the processing node MUST behave as specified by RFC 3209 [RFC3209] and drop the RRO from the Path message entirely. If the SRLG collection request was contained in an LSP_ATTRIBUTES Object, the processing node MAY omit some or all of the requested SRLGs from the RRO; otherwise it MUST behave as specified by [RFC3209] and drop the RRO from the Path message entirely. Subsequent processing of the LSP proceeds as further specified in RFC 3209 [RFC3209].

Following the steps described above, the intermediate nodes of the LSP can collect the SRLG information in the RRO during the processing of the Path message hop by hop. When the Path message arrives at the egress node, the egress node receives SRLG information in the RRO.

Per RFC 3209 [RFC3209], when issuing a Resv message for a Path message which contains an RRO, an egress node initiates the RRO process by adding an RRO to the outgoing Resv message. The processing for RROs contained in Resv messages then mirrors that of the Path messages.

When a node receives a Resv message for an LSP for which SRLG Collection was specified in the corresponding Path message, then when local policy allows recording SRLG information, the node MUST add SRLG information to the RRO of the corresponding outgoing Resv message as specified below. When the Resv message arrives at the ingress node, the ingress node can extract the SRLG information from the RRO in the same way as the egress node.

Note that a link’s SRLG information for the upstream direction cannot be assumed to be the same as that in the downstream.

- For Path and Resv messages for a unidirectional LSP, a node SHOULD include SRLG sub-objects in the RRO for the downstream data link only.

- For Path and Resv messages for a bidirectional LSP, a node SHOULD include SRLG sub-objects in the RRO for both the upstream data link and the downstream data link from the local node. In this case, the node MUST include the information in the same order for both Path messages and Resv messages. That is, the SRLG sub-object for the upstream link is added to the RRO before the SRLG

sub-object for the downstream link.

If SRLG data is added for both the upstream and downstream links, the two sets of SRLG data MUST be added in separate SRLG sub-objects. A single SRLG sub-object MUST NOT contain a mixture of upstream and downstream SRLGs. When adding a SRLG sub-object to an RRO, the D-bit MUST be set appropriately to indicate the direction of the SRLGs. If an SRLG ID applies in both directions, it SHOULD be added to both the upstream and downstream SRLG sub-objects.

Based on the above procedure, the endpoints can get the SRLG information automatically. Then the endpoints can for instance advertise it as a TE link to the routing instance based on the procedure described in [RFC6107] and configure the SRLG information of the Forwarding Adjacency (FA) automatically.

5.2. SRLG Update

When the SRLG information of a link is changed, the endpoints of LSPs using that link need to be made aware of the changes. When a change to the set of SRLGs associated with a link occurs, the procedures defined in Section 4.4.3 of RFC 3209 [RFC3209] MUST be used to refresh the SRLG information for each affected LSP if the SRLG change is to be communicated to other nodes according to the local node's policy.

5.3 Domain Boundaries

If mandated by local policy as specified by the network operator, a node MAY remove SRLG information from any RRO in a Path or Resv message being processed. It MAY add a summary of the removed SRLGs or map them to other SRLG values. However, this SHOULD NOT be done unless explicitly mandated by local policy.

5.4. Compatibility

A node that does not recognize the SRLG Collection Flag in the Attribute Flags TLV is expected to proceed as specified in RFC 5420 [RFC5420]. It is expected to pass the TLV on unaltered if it appears in a LSP_ATTRIBUTES object, or reject the Path message with the appropriate Error Code and Value if it appears in a LSP_REQUIRED_ATTRIBUTES object.

A node that does not recognize the SRLG RRO sub-object is expected to behave as specified in RFC 3209 [RFC3209]: unrecognized sub-objects are to be ignored and passed on unchanged.

6. Manageability Considerations

6.1. Policy Configuration

In a border node of inter-domain or inter-layer network, the following SRLG processing policy MUST be capable of being configured:

- Whether the node is allowed to participate in SRLG collection and notify changes to collected SRLG information to endpoint nodes as described in section 5.2.

- Whether the SRLG IDs of the domain or specific layer network can be exposed to the nodes outside the domain or layer network, or whether they SHOULD be summarized, mapped to values that are comprehensible to nodes outside the domain or layer network, or removed entirely as described in section 5.3.

A node using RFC 5553 [RFC5553] and PKS MAY apply the same policy.

6.2. Coherent SRLG IDs

In a multi-layer multi-domain scenario, SRLG IDs can be configured by different management entities in each layer/domain. In such scenarios, maintaining a coherent set of SRLG IDs is a key requirement in order to be able to use the SRLG information properly. Thus, SRLG IDs SHOULD be unique. Note that current procedure is targeted towards a scenario where the different layers and domains belong to the same operator, or to several coordinated administrative groups. Ensuring the aforementioned coherence of SRLG IDs is beyond the scope of this document.

Further scenarios, where coherence in the SRLG IDs cannot be guaranteed are out of the scope of the present document and are left for further study.

7. Security Considerations

This document builds on the mechanisms defined in [RFC3473], which also discusses related security measures. In addition, [RFC5920] provides an overview of security vulnerabilities and protection mechanisms for the GMPLS control plane. The procedures defined in this document permit the transfer of SRLG data between layers or domains during the signaling of LSPs, subject to policy at the layer or domain boundary. As described in section 5.3 and section 6.1, local policy as specified by the network operator will explicitly mandate the processing of information at domain or layer boundaries.

8. IANA Considerations

8.1. RSVP Attribute Bit Flags

IANA has created a registry and manages the space of the Attribute bit flags of the Attribute Flags TLV, as described in section 11.3 of RFC 5420 [RFC5420], in the "Attribute Flags" section of the "Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Parameters" registry located in http://www.iana.org/assignments/rsvp-te-parameters".

This document introduces a new Attribute Bit Flag:

Bit No	Name	Attribute Flags Path	Attribute Flags Resv	ERO	RRO	Reference
TBD; suggested value: 12	SRLG Collection Flag	Yes	No	No	Yes	This I-D

8.2. ROUTE_RECORD Object

IANA manages the "RSVP PARAMETERS" registry located at http://www.iana.org/assignments/rsvp-parameters. This document introduces a new RRO sub-object:

Value	Description	Reference
TBD; suggested value: 34	SRLG sub-object	This I-D

8.3. Policy Control Failure Error subcodes

IANA manages the assignments in the "Error Codes and Globally-Defined Error Value Sub-Codes" section of the "RSVP PARAMETERS" registry located at http://www.iana.org/assignments/rsvp-parameters.

This document introduces a new Policy Control Failure Error sub-code:

Value	Description	Reference
TBD; suggested value: 21	SRLG Recording Rejected	This I-D

9. Contributors

Dan Li
Huawei
F3-5-B RD Center
Bantian, Longgang District, Shenzhen 518129

P.R.China
Email: danli@huawei.com

10. Acknowledgements

The authors would like to thank Dieter Beller, Vishnu Pavan Beeram, Lou Berger, Deborah Brungard, Igor Bryskin, Ramon Casellas, Niclas Comstedt, Alan Davey, Elwyn Davies, Dhruv Dhody, Himanshu Shah and Xian Zhang for their useful comments and improvements to this document.

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.

[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.

[RFC4202] Kompella, K. and Y. Rekhter, "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, October 2005.

[RFC5420] Farrel, A., Papadimitriou, D., Vasseur, JP., and A. Ayyangarps, "Encoding of Attributes for MPLS LSP Establishment Using Resource Reservation Protocol Traffic Engineering (RSVP-TE)", RFC 5420, February 2009.

[RFC5520] Bradford, R., Vasseur, JP., and A. Farrel, "Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism", RFC 5520, April 2009.

[RFC5553] Farrel, A., Bradford, R., and JP. Vasseur, "Resource Reservation Protocol (RSVP) Extensions for Path Key Support", RFC 5553, May 2009.

11.2. Informative References

[RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching

(GMPLS) Traffic Engineering (TE)", RFC 4206, October 2005.

[RFC4208] Swallow, G., Drake, J., Ishimatsu, H., and Y. Rekhter, "Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model", RFC 4208, October 2005.

[RFC4874] Lee, CY., Farrel, A., and S. De Cnodder, "Exclude Routes - Extension to Resource ReserVation Protocol-Traffic Engineering (RSVP-TE)", RFC 4874, April 2007.

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

[RFC6107] Shiomoto, K. and A. Farrel, "Procedures for Dynamically Signaled Hierarchical Label Switched Paths", RFC 6107, February 2011.

Authors' Addresses

Fatai Zhang (editor)
Huawei
F3-5-B RD Center
Bantian, Longgang District, Shenzhen 518129
P.R.China
Email: zhangfatai@huawei.com

Oscar Gonzalez de Dios (editor)
Telefonica Global CTO
Distrito Telefonica, edificio sur, Ronda de la Comunicacion 28045
Madrid 28050
Spain
Phone: +34 913129647
Email: oscar.gonzalezdedios@telefonica.com

Cyril Margaria
Suite 4001, 200 Somerset Corporate Blvd.
Bridgewater, NJ 08807
US
Email: cyril.margaria@gmail.com

Matt Hartley
Cisco
Email: mhartley@cisco.com

Zafar Ali
Cisco
Email: zali@cisco.com

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 22, 2019

V. Beeram
Juniper Networks
T. Saad, Ed.
R. Gandhi
Cisco Systems, Inc.
X. Liu
Jabil
I. Bryskin
Huawei Technologies
H. Shah
Ciena

February 18, 2019

A YANG Data Model for Resource Reservation Protocol (RSVP)
draft-ietf-teas-yang-rsvp-10

Abstract

This document defines a YANG data model for the configuration and management of RSVP Protocol. The model covers the building blocks of the RSVP protocol that can be augmented and used by other RSVP extension models such as RVSP extensions to Traffic-Engineering (RSVP-TE). The model covers the configuration, operational state, remote procedural calls, and event notifications data.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on August 22, 2019.

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 interfaces, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage the RSVP protocol [RFC2205]. This model covers RSVP protocol building blocks that can be augmented and used by other RSVP extension models- such as for signaling RSVP-TE MPLS (or other technology specific) Label Switched Paths (LSP)s.

1.1. Terminology

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

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

1.2. Model Tree Diagram

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

1.3. Prefixes in Data Node Names

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

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
rt-type	ietf-routing-types	XX
key-chain	ietf-key-chain	XX

Table 1: Prefixes and corresponding YANG modules

2. Model Overview

The RSVP base YANG module augments the "control-plane-protocol" list in ietf-routing [RFC8349] module with specific RSVP parameters in an "rsvp" container. It also defines an extensiion identity "rsvp" of base "rt:routing-protocol" to identify the RSVP protocol.

The augmentation of the RSVP model by other models (e.g. RSVP-TE for MPLS or other technologies) are outside the scope of this document and are discussed in separate document(s), e.g. [I-D.ietf-teas-yang-rsvp-te].

2.1. Module(s) Relationship

This document divides the RSVP model into two modules: base and extended RSVP modules. Some RSVP features are categorized as core to the function of the protocol that are supported by most vendors claiming support for RSVP protocol. Such features configuration and state are grouped in the RSVP base module.

Other extended RSVP features are categorized as either optional or providing knobs to better tune basic functionality of the RSVP protocol. The support for extended RSVP features by all vendors is considered optional. Such features are grouped in a separate RSVP extended module.

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

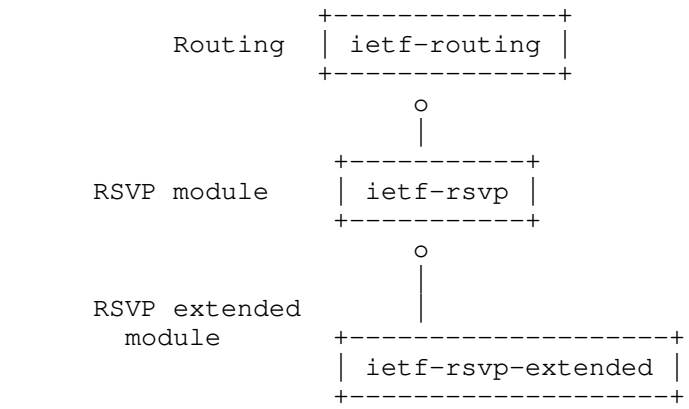


Figure 1: Relationship of RSVP and RSVP extended modules with other protocol modules

2.2. Design Considerations

The RSVP base model does not aim to be feature complete. The primary intent is to cover a set of standard core features that are commonly in use. For example:

- Authentication ([RFC2747])
- Refresh Reduction ([RFC2961])
- Hellos ([RFC3209])
- Graceful Restart ([RFC3473], [RFC5063])

The extended RSVP YANG model covers non-basic configuration(s) for RSVP feature(s) as well as optional RSVP feature that are not a must for basic RSVP operation.

The defined data model supports configuration inheritance for neighbors, and interfaces. Data elements defined in the main container (e.g. the container that encompasses the list of interfaces, or neighbors) are assumed to apply equally to all elements of the list, unless overridden explicitly for a certain element (e.g. interface). Vendors are expected to augment the above container(s) to provide the list of inheritance command for their implementations.

2.3. RSVP Base YANG Model

The RSVP base YANG data model defines the container "rsvp" as the top level container in this data model. The presence of this container enables the RSVP protocol functionality.

Derived state data is contained under a "state" container of the intended object as shown in Figure 2.

```
module: ietf-rsvp
   +--rw rsvp!
      +--rw globals
         .
         .
      +--rw interfaces
            .
            +-- ro state
               <<derived state associated with interfaces>>
         .
         .
      +--rw neighbors
            .
            +-- ro state
               <<derived state associated with the tunnel>>
         .
         .
      +--rw sessions
            .
            +-- ro state
               <<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 high-level tree model view

The following subsections provide overview of the parts of the model pertaining to configuration and state data.

Configuration and state data are organized into those applicable globally (node scope), per interface, per neighbor, or per session.

Global Data:

The global data branch of the model covers configuration and state that are applicable the RSVP protocol behavior.

Interface Data:

The interface data branch of the data model covers configuration and state elements relevant to one or all RSVP interfaces. Any data configuration applied at the "interfaces" container level are equally applicable to all interfaces - unless overridden by explicit configuration under a specific interface.

Neighbor Data:

The neighbor data branch of the data model covers configuration and state elements relevant to RSVP neighbors.

Session Data:

The sessions data branch covers configuration of elements relevant to RSVP sessions.

2.3.1. Tree Diagram

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

```
 module: ietf-rsvp
   augment /rt:routing/rt:control-plane-protocols
             /rt:control-plane-protocol:
     +--rw rsvp!
        +--rw globals
        |  +--rw sessions
        |  |  +--ro session* [local-index]
        |  |     +--ro local-index    -> ../state/local-index
        |  |     +--ro state
        |  |        +--ro local-index?        uint64
        |  |        +--ro destination-port?   inet:port-number
        |  |        +--ro source?             inet:ip-address
        |  |        +--ro destination?        inet:ip-address
        |  |        +--ro session-name?       string
        |  |        +--ro session-state?      enumeration
        |  |        +--ro session-type?       identityref
        |  |        +--ro psbs
        |  |        |  +--ro psb* []
        |  |        |     +--ro source-port?   inet:port-number
        |  |        |     +--ro expires-in?    uint32
        |  |        +--ro rsbs
        |  |           +--ro rsb* []
        |  |              +--ro source-port?         inet:port-number
        |  |              +--ro reservation-style?   identityref
        |  |              +--ro expires-in?          uint32
        |  +--rw statistics
        |  |  +--ro state
```

```
|  |  |     +--ro messages
|  |  |     |  +--ro ack-sent?                          yang:counter64
|  |  |     |  +--ro ack-received?                      yang:counter64
|  |  |     |  +--ro bundle-sent?                       yang:counter64
|  |  |     |  +--ro bundle-received?                   yang:counter64
|  |  |     |  +--ro hello-sent?                        yang:counter64
|  |  |     |  +--ro hello-received?                    yang:counter64
|  |  |     |  +--ro integrity-challenge-sent?          yang:counter64
|  |  |     |  +--ro integrity-challenge-received?      yang:counter64
|  |  |     |  +--ro integrity-response-sent?           yang:counter64
|  |  |     |  +--ro integrity-response-received?       yang:counter64
|  |  |     |  +--ro notify-sent?                       yang:counter64
|  |  |     |  +--ro notify-received?                   yang:counter64
|  |  |     |  +--ro path-sent?                         yang:counter64
|  |  |     |  +--ro path-received?                     yang:counter64
|  |  |     |  +--ro path-err-sent?                     yang:counter64
|  |  |     |  +--ro path-err-received?                 yang:counter64
|  |  |     |  +--ro path-tear-sent?                    yang:counter64
|  |  |     |  +--ro path-tear-received?                yang:counter64
|  |  |     |  +--ro resv-sent?                         yang:counter64
|  |  |     |  +--ro resv-received?                     yang:counter64
|  |  |     |  +--ro resv-confirm-sent?                 yang:counter64
|  |  |     |  +--ro resv-confirm-received?             yang:counter64
|  |  |     |  +--ro resv-err-sent?                     yang:counter64
|  |  |     |  +--ro resv-err-received?                 yang:counter64
|  |  |     |  +--ro resv-tear-sent?                    yang:counter64
|  |  |     |  +--ro resv-tear-received?                yang:counter64
|  |  |     |  +--ro summary-refresh-sent?              yang:counter64
|  |  |     |  +--ro summary-refresh-received?          yang:counter64
|  |  |     |  +--ro unknown-messages-received?         yang:counter64
|  |  |     +--ro packets
|  |  |     |  +--ro sent?       yang:counter64
|  |  |     |  +--ro received?   yang:counter64
|  |  |     +--ro errors
|  |  |        +--ro authenticate?   yang:counter64
|  |  |        +--ro checksum?       yang:counter64
|  |  |        +--ro packet-len?     yang:counter64
|  |  +--rw graceful-restart
|  |     +--rw enabled?   boolean
|  +--rw interfaces
|  |  +--rw refresh-reduction
|  |  |  +--rw enabled?   boolean
|  |  +--rw hellos
|  |  |  +--rw enabled?   boolean
|  |  +--rw authentication
|  |  |  +--rw enabled?              boolean
|  |  |  +--rw authentication-key?   string
|  |  |  +--rw crypto-algorithm      identityref
```

```
|  +--rw statistics
|  |  +--ro state
|  |     +--ro messages
|  |     |  +--ro ack-sent?                          yang:counter64
|  |     |  +--ro ack-received?                      yang:counter64
|  |     |  +--ro bundle-sent?                       yang:counter64
|  |     |  +--ro bundle-received?                   yang:counter64
|  |     |  +--ro hello-sent?                        yang:counter64
|  |     |  +--ro hello-received?                    yang:counter64
|  |     |  +--ro integrity-challenge-sent?          yang:counter64
|  |     |  +--ro integrity-challenge-received?      yang:counter64
|  |     |  +--ro integrity-response-sent?           yang:counter64
|  |     |  +--ro integrity-response-received?       yang:counter64
|  |     |  +--ro notify-sent?                       yang:counter64
|  |     |  +--ro notify-received?                   yang:counter64
|  |     |  +--ro path-sent?                         yang:counter64
|  |     |  +--ro path-received?                     yang:counter64
|  |     |  +--ro path-err-sent?                     yang:counter64
|  |     |  +--ro path-err-received?                 yang:counter64
|  |     |  +--ro path-tear-sent?                    yang:counter64
|  |     |  +--ro path-tear-received?                yang:counter64
|  |     |  +--ro resv-sent?                         yang:counter64
|  |     |  +--ro resv-received?                     yang:counter64
|  |     |  +--ro resv-confirm-sent?                 yang:counter64
|  |     |  +--ro resv-confirm-received?             yang:counter64
|  |     |  +--ro resv-err-sent?                     yang:counter64
|  |     |  +--ro resv-err-received?                 yang:counter64
|  |     |  +--ro resv-tear-sent?                    yang:counter64
|  |     |  +--ro resv-tear-received?                yang:counter64
|  |     |  +--ro summary-refresh-sent?              yang:counter64
|  |     |  +--ro summary-refresh-received?          yang:counter64
|  |     |  +--ro unknown-messages-received?         yang:counter64
|  |     +--ro packets
|  |     |  +--ro sent?       yang:counter64
|  |     |  +--ro received?   yang:counter64
|  |     +--ro errors
|  |        +--ro authenticate?   yang:counter64
|  |        +--ro checksum?       yang:counter64
|  |        +--ro packet-len?     yang:counter64
|  +--rw interface* [interface]
|     +--rw interface            if:interface-ref
|     +--rw refresh-reduction
|     |  +--rw enabled?   boolean
|     +--rw hellos
|     |  +--rw enabled?   boolean
|     +--rw authentication
|     |  +--rw enabled?              boolean
|     |  +--rw authentication-key?   string
```

```
|      |  +--rw crypto-algorithm      identityref
|      +--rw statistics
|         +--ro state
|            +--ro messages
|               |  +--ro ack-sent?
|               |  |       yang:counter64
|               |  +--ro ack-received?
|               |  |       yang:counter64
|               |  +--ro bundle-sent?
|               |  |       yang:counter64
|               |  +--ro bundle-received?
|               |  |       yang:counter64
|               |  +--ro hello-sent?
|               |  |       yang:counter64
|               |  +--ro hello-received?
|               |  |       yang:counter64
|               |  +--ro integrity-challenge-sent?
|               |  |       yang:counter64
|               |  +--ro integrity-challenge-received?
|               |  |       yang:counter64
|               |  +--ro integrity-response-sent?
|               |  |       yang:counter64
|               |  +--ro integrity-response-received?
|               |  |       yang:counter64
|               |  +--ro notify-sent?
|               |  |       yang:counter64
|               |  +--ro notify-received?
|               |  |       yang:counter64
|               |  +--ro path-sent?
|               |  |       yang:counter64
|               |  +--ro path-received?
|               |  |       yang:counter64
|               |  +--ro path-err-sent?
|               |  |       yang:counter64
|               |  +--ro path-err-received?
|               |  |       yang:counter64
|               |  +--ro path-tear-sent?
|               |  |       yang:counter64
|               |  +--ro path-tear-received?
|               |  |       yang:counter64
|               |  +--ro resv-sent?
|               |  |       yang:counter64
|               |  +--ro resv-received?
|               |  |       yang:counter64
|               |  +--ro resv-confirm-sent?
|               |  |       yang:counter64
|               |  +--ro resv-confirm-received?
|               |  |       yang:counter64
```

```
|            |  +--ro resv-err-sent?
|            |  |       yang:counter64
|            |  +--ro resv-err-received?
|            |  |       yang:counter64
|            |  +--ro resv-tear-sent?
|            |  |       yang:counter64
|            |  +--ro resv-tear-received?
|            |  |       yang:counter64
|            |  +--ro summary-refresh-sent?
|            |  |       yang:counter64
|            |  +--ro summary-refresh-received?
|            |  |       yang:counter64
|            |  +--ro unknown-messages-received?
|            |          yang:counter64
|            +--ro packets
|            |  +--ro sent?       yang:counter64
|            |  +--ro received?   yang:counter64
|            +--ro errors
|               +--ro authenticate?   yang:counter64
|               +--ro checksum?       yang:counter64
|               +--ro packet-len?     yang:counter64
+--rw neighbors
   +--rw neighbor* [address]
      +--rw address    inet:ip-address
      +--ro state
         +--ro address?                      inet:ip-address
         +--ro epoch?                        uint32
         +--ro expiry-time?                  uint32
         +--ro graceful-restart
         |  +--ro enabled?                   boolean
         |  +--ro local-restart-time?        uint32
         |  +--ro local-recovery-time?       uint32
         |  +--ro neighbor-restart-time?     uint32
         |  +--ro neighbor-recovery-time?    uint32
         |  +--ro helper-mode
         |     +--ro enabled?                                  boolean
         |     +--ro max-helper-restart-time?                  uint32
         |     +--ro max-helper-recovery-time?                 uint32
         |     +--ro neighbor-restart-time-remaining?          uint32
         |     +--ro neighbor-recovery-time-remaining?         uint32
         +--ro hello-status?                 enumeration
         +--ro interface?                    if:interface-ref
         +--ro neighbor-state?               enumeration
         +--ro refresh-reduction-capable?    boolean
         +--ro restart-count?                yang:counter32
         +--ro restart-time?                 yang:date-and-time
```

Figure 3: RSVP model tree diagram

2.3.2. YANG Module

```
<CODE BEGINS> file "ietf-rsvp@2019-02-18.yang"
module ietf-rsvp {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp";

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

  import ietf-interfaces {
    prefix if;
    reference "RFC8343: A YANG Data Model for Interface Management";
  }

  import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC8349: A YANG Data Model for Routing Management
       (NMDA Version)";
  }

  import ietf-key-chain {
    prefix "key-chain";
    reference "RFC8177: YANG Data Model for Key Chains";
  }

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

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

     WG Chair: Lou Berger
               <mailto:lberger@labn.net>
```

```
        WG Chair: Vishnu Pavan Beeram
                  <mailto:vbeeram@juniper.net>

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

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

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

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

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

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

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

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

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

     // RFC Ed.: replace XXXX with actual RFC number and remove this
     // note.

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

     revision "2019-02-18" {
       description
         "A YANG Data Model for Resource Reservation Protocol";
       reference
```

```
        "RFCXXXX: A YANG Data Model for Resource Reservation Protocol
         (RSVP)";
    }

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

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

    identity rsvp-session-ipv4 {
      base rsvp-session-type;
      description "RSVP IPv4 session type";
    }

    identity rsvp-session-ipv6 {
      base rsvp-session-type;
      description "RSVP IPv4 session type";
    }

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

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

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

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

    grouping graceful-restart-config {
      description
        "Base configuration parameters relating to RSVP
```

```
          Graceful-Restart";
        leaf enabled {
          type boolean;
          description
            "'true' if RSVP Graceful Restart is enabled.
             'false' if RSVP Graceful Restart is disabled.";
        }
      }

      grouping graceful-restart {
        description
          "RSVP graceful restart parameters grouping";
        container graceful-restart {
          description
            "RSVP graceful restart parameters container";
          uses graceful-restart-config;
        }
      }

      grouping refresh-reduction-config {
        description
          "Configuration parameters relating to RSVP
           refresh reduction";

        leaf enabled {
          type boolean;
          description
            "'true' if RSVP Refresh Reduction is enabled.
             'false' if RSVP Refresh Reduction is disabled.";
        }
      }

      grouping refresh-reduction {
        description
          "Top level grouping for RSVP refresh reduction
           parameters";
        container refresh-reduction {
          description
            "Top level container for RSVP refresh reduction
             parameters";
          uses refresh-reduction-config;
        }
      }

      grouping authentication-config {
        description
          "Configuration parameters relating to RSVP
           authentication";
```

```
      leaf enabled {
        type boolean;
        description
          "'true' if RSVP Authentication is enabled.
           'false' if RSVP Authentication is disabled.";
      }
      leaf authentication-key {
        type string;
        description
          "An authentication key string";
        reference
          "RFC 2747: RSVP Cryptographic Authentication";
      }
      leaf crypto-algorithm {
       type identityref {
         base key-chain:crypto-algorithm;
       }
       mandatory true;
       description
         "Cryptographic algorithm associated with key.";
      }
    }

    grouping authentication {
      description
        "Top level grouping for RSVP authentication parameters";
      container authentication {
        description
          "Top level container for RSVP authentication
           parameters";
        uses authentication-config;
      }
    }

    grouping hellos-config {
      description
        "Configuration parameters relating to RSVP
        hellos";
      leaf enabled {
        type boolean;
        description
          "'true' if RSVP Hello is enabled.
           'false' if RSVP Hello is disabled.";
      }
    }

    grouping hellos {
      description
```

```
        "Top level grouping for RSVP hellos parameters";
      container hellos {
        description
          "Top level container for RSVP hello parameters";
        uses hellos-config;
      }
    }

    grouping signaling-parameters-config {
      description
        "Configuration parameters relating to RSVP
        signaling";
    }

    grouping signaling-parameters {
      description
        "Top level grouping for RSVP signaling  parameters";
      uses signaling-parameters-config;
    }

    grouping session-attributes-state {
      description
        "Top level grouping for RSVP session properties";
      leaf local-index {
        type uint64;
        description
          "The index used to identify the RSVP session
           on the local network element. This index is
           generated by the device and is unique only
           to the local network element.";
      }
      leaf destination-port {
        type inet:port-number;
        description "RSVP destination port";
        reference "RFC2205";
      }
      leaf source {
        type inet:ip-address;
        description "RSVP source address";
        reference "RFC2205";
      }
      leaf destination {
        type inet:ip-address;
        description "RSVP destination address";
        reference "RFC2205";
      }
      leaf session-name {
        type string;
```

```
        description
          "The signaled name of this RSVP session.";
      }
      leaf session-state {
        type enumeration {
          enum "up" {
            description
              "RSVP session is up";
          }
          enum "down" {
            description
              "RSVP session is down";
          }
        }
        description
          "Enumeration of RSVP session states";
      }
      leaf session-type {
        type identityref {
          base rsvp-session-type;
        }
        description "RSVP session type";
      }
      container psbs {
        description "Path State Block container";
        list psb {
          description "List of path state blocks";
          leaf source-port {
            type inet:port-number;
            description "RSVP source port";
            reference "RFC2205";
          }
          leaf expires-in {
            type uint32;
            units seconds;
            description "Time to reservation expiry (in seconds)";
          }
        }
      }
      container rsbs {
        description "Reservation State Block container";
        list rsb {
          description "List of reservation state blocks";
          leaf source-port {
            type inet:port-number;
            description "RSVP source port";
            reference "RFC2205";
          }
```

```
          leaf reservation-style {
            type identityref {
              base reservation-style;
            }
            description "RSVP reservation style";
          }
          leaf expires-in {
            type uint32;
            units seconds;
            description "Time to reservation expiry (in seconds)";
          }
        }
      }
    }

    grouping neighbor-attributes {
      description
        "Top level grouping for RSVP neighbor properties";
      leaf address {
        type inet:ip-address;
        description
          "Address of RSVP neighbor";
      }
      container state {
        config false;
        description
          "State information associated with RSVP
           neighbor properties";
        uses neighbor-derived-state;
      }
    }

    grouping packets-state {
      description
        "Packet statistics grouping";
      container packets {
        description
          "Packet statistics container";
        leaf sent {
          type yang:counter64;
          description
            "Packet sent count";
        }

        leaf received {
          type yang:counter64;
          description
            "Packet received count";
```

```
        }
      }
    }

    grouping protocol-state {
      description
        "RSVP protocol statistics grouping";
      container messages {
        description
          "RSVP protocol statistics container";
        leaf ack-sent {
          type yang:counter64;
          description
            "Hello sent count";
        }

        leaf ack-received {
          type yang:counter64;
          description
            "Hello received count";
        }

        leaf bundle-sent {
          type yang:counter64;
          description
            "Bundle sent count";
        }

        leaf bundle-received {
          type yang:counter64;
          description
            "Bundle received count";
        }

        leaf hello-sent {
          type yang:counter64;
          description
            "Hello sent count";
        }

        leaf hello-received {
          type yang:counter64;
          description
            "Hello received count";
        }

        leaf integrity-challenge-sent {
          type yang:counter64;
```

```
          description
            "Integrity Challenge sent count";
        }

        leaf integrity-challenge-received {
          type yang:counter64;
          description
            "Integrity Challenge received count";
        }

        leaf integrity-response-sent {
          type yang:counter64;
          description
            "Integrity Response sent count";
        }

        leaf integrity-response-received {
          type yang:counter64;
          description
            "Integrity Response received count";
        }

        leaf notify-sent {
          type yang:counter64;
          description
            "Notify sent count";
        }

        leaf notify-received {
          type yang:counter64;
          description
            "Notify received count";
        }

        leaf path-sent {
          type yang:counter64;
          description
            "Path sent count";
        }

        leaf path-received {
          type yang:counter64;
          description
            "Path received count";
        }

        leaf path-err-sent {
          type yang:counter64;
```

```
          description
            "Path error sent count";
        }

        leaf path-err-received {
          type yang:counter64;
          description
            "Path error received count";
        }

        leaf path-tear-sent {
          type yang:counter64;
          description
            "Path tear sent count";
        }

        leaf path-tear-received {
          type yang:counter64;
          description
            "Path tear received count";
        }

        leaf resv-sent {
          type yang:counter64;
          description
            "Resv sent count";
        }

        leaf resv-received {
          type yang:counter64;
          description
            "Resv received count";
        }

        leaf resv-confirm-sent {
          type yang:counter64;
          description
            "Confirm sent count";
        }

        leaf resv-confirm-received {
          type yang:counter64;
          description
            "Confirm received count";
        }

        leaf resv-err-sent {
          type yang:counter64;
```

```
          description
            "Resv error sent count";
        }

        leaf resv-err-received {
          type yang:counter64;
          description
            "Resv error received count";
        }

        leaf resv-tear-sent {
          type yang:counter64;
          description
            "Resv tear sent count";
        }

        leaf resv-tear-received {
          type yang:counter64;
          description
            "Resv tear received count";
        }

        leaf summary-refresh-sent {
          type yang:counter64;
          description
            "Summary refresh sent count";
        }

        leaf summary-refresh-received {
          type yang:counter64;
          description
            "Summary refresh received count";
        }

        leaf unknown-messages-received {
          type yang:counter64;
          description
            "Unknown packet received count";
        }
      }
    }

    grouping errors-state {
      description
        "Error statistics state grouping";
      container errors {
        description
          "Error statistics state container";
```

```
        leaf authenticate {
          type yang:counter64;
          description
            "The total number of packets received with an
            authentication failure.";
        }

        leaf checksum {
          type yang:counter64;
          description
            "The total number of packets received with an invalid
            checksum value.";
        }

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

    grouping statistics-state {
      description "RSVP statistic attributes.";
      container statistics {
        description
          "statistics state container";
        container state {
          config false;
          description
            "State information associated with RSVP
             hello parameters";
          uses protocol-state;
          uses packets-state;
          uses errors-state;
        }
      }
    }

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

      leaf address {
        type inet:ip-address;
        description
          "Address of RSVP neighbor";
```

```
      }

      leaf epoch {
        type uint32;
        description
          "Neighbor epoch.";
      }

      leaf expiry-time {
        type uint32;
        units seconds;
        description
          "Neighbor expiry time after which the neighbor state
          is purged if no states associated with it";
      }

      container graceful-restart {
        description
          "Graceful restart information.";

        leaf enabled {
          type boolean;
          description
            "'true' if graceful restart is enabled for the neighbor.";
        }

        leaf local-restart-time {
          type uint32;
          units seconds;
          description
            "Local node restart time";
        }

        leaf local-recovery-time {
          type uint32;
          units seconds;
          description
            "Local node recover time";
        }

        leaf neighbor-restart-time {
          type uint32;
          units seconds;
          description
            "Neighbor restart time";
        }

        leaf neighbor-recovery-time {
```

```
          type uint32;
          units seconds;
          description
            "Neighbor recover time";
        }

        container helper-mode {
          description
            "Helper mode information ";

          leaf enabled {
            type boolean;
            description
              "'true' if helper mode is enabled.";
          }

          leaf max-helper-restart-time {
            type uint32;
            units seconds;
            description
              "The time the router or switch waits after it
               discovers that a neighboring router has gone down
               before it declares the neighbor down";
          }

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

          leaf neighbor-restart-time-remaining {
            type uint32;
            units seconds;
            description
              "Number of seconds remaining for neighbor to send
               Hello message after restart.";
          }

          leaf neighbor-recovery-time-remaining {
            type uint32;
            units seconds;
            description
              "Number of seconds remaining for neighbor to
               refresh.";
          }
```

```
        } // helper-mode
      } // graceful-restart

      leaf hello-status {
        type enumeration {
          enum "enabled" {
            description
              "Enabled";
          }
          enum "disabled" {
            description
              "Disabled";
          }
          enum "restarting" {
            description
              "Restarting";
          }
        }
        description
          "Hello status";
      }

      leaf interface {
        type if:interface-ref;
        description
          "Interface where RSVP neighbor was detected";
      }

      leaf neighbor-state {
        type enumeration {
          enum "up" {
            description
              "up";
          }
          enum "down" {
            description
              "down";
          }
          enum "hello-disable" {
            description
              "hello-disable";
          }
          enum "restarting" {
            description
              "restarting";
          }
        }
        description
```

```
          "Neighbor state";
      }

      leaf refresh-reduction-capable {
        type boolean;
        description
          "enables all RSVP refresh reduction message
           bundling, RSVP message ID, reliable message delivery
           and summary refresh";
        reference
          "RFC 2961 RSVP Refresh Overhead Reduction
           Extensions";
      }

      leaf restart-count {
        type yang:counter32;
        description
          "Number of times this neighbor restart";
      }

      leaf restart-time {
        type yang:date-and-time;
        description
          "Last restart time of the neighbor";
      }
    }

    grouping global-attributes {
      description
        "Top level grouping for RSVP global properties";
      container sessions {
        description
          "RSVP sessions container";
        list session {
          key "local-index";
          config false;
          description
            "List of RSVP sessions";

          leaf local-index {
            type leafref {
              path "../state/local-index";
            }
            description
              "Reference to the local index for the RSVP
              session";
          }
          container state {
```

```
            config false;
            description
              "State information associated with RSVP
               session parameters";
            uses session-attributes-state;
          }
        }
      }
      uses statistics-state;
    }

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

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol" {
      when "rt:type = 'rsvp:rsvp'" {
      description
       "This augment is only valid when routing protocol
        instance type is RSVP.";
      }
      description
        "RSVP protocol augmentation";
      container rsvp {
        presence "Enable RSVP feature";
        description "RSVP feature container";
        container globals {
          description "RSVP global properties.";
          uses global-attributes;
          uses graceful-restart;
        }

        container interfaces {
          description
            "RSVP interfaces container";
          uses intf-attributes;

          list interface {
            key "interface";
            description
              "RSVP interfaces.";
```

```
            leaf interface {
              type if:interface-ref;
              description
                "RSVP interface.";
            }
            uses intf-attributes;
          }
        }

        container neighbors {
          description "RSVP neighbors container";
          list neighbor {
            key "address";
            description "List of RSVP neighbors";
            uses neighbor-attributes;
          }
        }
      }
    }
  }
  <CODE ENDS>
```

2.4. RSVP Extended YANG Model

The RSVP extended YANG model covers non-core RSVP feature(s). It also covers feature(s) that are not necessarily supported by all vendors, and hence, can be guarded with "if-feature" checks.

2.4.1. Tree Diagram

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

```
module: ietf-rsvp-extended
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
            /rsvp:graceful-restart:
    +--rw restart-time?    uint32
    +--rw recovery-time?   uint32
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
            /rsvp:statistics/rsvp:state/rsvp:packets:
    +--ro discontinuity-time?   yang:date-and-time
    +--ro out-dropped?          yang:counter64
    +--ro in-dropped?           yang:counter64
    +--ro out-error?            yang:counter64
    +--ro in-error?             yang:counter64
  augment /rt:routing/rt:control-plane-protocols
```

```
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
            /rsvp:statistics/rsvp:state/rsvp:messages:
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
            /rsvp:statistics/rsvp:state/rsvp:errors:
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces:
       +--rw refresh-interval?        uint32
       +--rw refresh-misses?          uint32
       +--rw checksum?                boolean
       +--rw patherr-state-removal?   empty
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
            /rsvp:refresh-reduction:
       +--rw bundle-message-max-size?    uint32
       +--rw reliable-ack-hold-time?     uint32
       +--rw reliable-ack-max-size?      uint32
       +--rw reliable-retransmit-time?   uint32
       +--rw reliable-srefresh?          empty
       +--rw summary-max-size?           uint32
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
            /rsvp:hellos:
       +--rw interface-based?   empty
       +--rw hello-interval?    uint32
       +--rw hello-misses?      uint32
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
            /rsvp:authentication:
       +--rw lifetime?      uint32
       +--rw window-size?   uint32
       +--rw challenge?     empty
       +--rw retransmits?   uint32
       +--rw key-chain?     key-chain:key-chain-ref
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
            /rsvp:interface:
       +--rw refresh-interval?        uint32
       +--rw refresh-misses?          uint32
       +--rw checksum?                boolean
       +--rw patherr-state-removal?   empty
     augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
            /rsvp:interface/rsvp:refresh-reduction:
       +--rw bundle-message-max-size?    uint32
       +--rw reliable-ack-hold-time?     uint32
       +--rw reliable-ack-max-size?      uint32
       +--rw reliable-retransmit-time?   uint32
```

```
      +--rw reliable-srefresh?          empty
      +--rw summary-max-size?           uint32
    augment /rt:routing/rt:control-plane-protocols
              /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
              /rsvp:interface/rsvp:hellos:
      +--rw interface-based?   empty
      +--rw hello-interval?    uint32
      +--rw hello-misses?      uint32
    augment /rt:routing/rt:control-plane-protocols
              /rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces
              /rsvp:interface/rsvp:authentication:
      +--rw lifetime?      uint32
      +--rw window-size?   uint32
      +--rw challenge?     empty
      +--rw retransmits?   uint32
      +--rw key-chain?     key-chain:key-chain-ref
```

Figure 4: RSVP extended model tree diagram

2.4.2. YANG Module

Figure 5 shows the RSVP extended YANG module:

```
<CODE BEGINS> file "ietf-rsvp-extended@2019-02-18.yang"
module ietf-rsvp-extended {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-rsvp-extended";

  prefix "rsvp-ext";

  import ietf-rsvp {
    prefix "rsvp";
    reference
      "RFCXXXX:  A YANG Data Model for Resource Reservation Protocol
       (RSVP)";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC8349: A YANG Data Model for Routing Management
       (NMDA Version)";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
```

```
    }

    import ietf-key-chain {
      prefix "key-chain";
      reference "RFC8177: YANG Data Model for Key Chains";
    }

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

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

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

       WG Chair: Lou Berger
                 <mailto:lberger@labn.net>

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

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

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

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

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

       Editor:   Xufeng Liu
                 <mailto:Xufeng_Liu@jabil.com>

       Editor:   Xia Chen
                 <mailto:jescia.chenxia@huawei.com>

       Editor:   Raqib Jones
                 <mailto:raqib@Brocade.com>

       Editor:   Bin Wen
                 <mailto:Bin_Wen@cable.comcast.com>";

    description
      "This module contains the Extended RSVP YANG data model.
```

```
        The model fully conforms to the Network Management Datastore
        Architecture (NMDA).

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

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

      // RFC Ed.: replace XXXX with actual RFC number and remove this
      // note.

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

      revision "2019-02-18" {
        description
          "A YANG Data Model for Extended Resource Reservation
           Protocol";
        reference
          "RFCXXXX: A YANG Data Model for Extended Resource Reservation
           Protocol (RSVP)";
      }

      /* RSVP features */
      feature authentication {
        description
          "Indicates support for RSVP authentication";
      }

      feature error-statistics {
        description
          "Indicates support for error statistics";
      }

      feature global-statistics {
        description
          "Indicates support for global statistics";
      }

      feature graceful-restart {
        description
```

```
        "Indicates support for RSVP graceful restart";
    }

    feature hellos {
      description
        "Indicates support for RSVP hellos (RFC3209).";
    }

    feature notify {
      description
        "Indicates support for RSVP notify message (RFC3473).";
    }

    feature refresh-reduction {
      description
        "Indicates support for RSVP refresh reduction (RFC2961).";
    }

    feature refresh-reduction-extended {
      description
        "Indicates support for RSVP refresh reduction (RFC2961).";
    }

    feature per-interface-statistics {
      description
        "Indicates support for per interface statistics";
    }

    grouping graceful-restart-extended-config {
      description
        "Configuration parameters relating to RSVP
         Graceful-Restart";
      leaf restart-time {
        type uint32;
        units seconds;
        description
          "Graceful restart time (seconds).";
        reference
          "RFC 5495: Description of the Resource
           Reservation Protocol - Traffic-Engineered
           (RSVP-TE) Graceful Restart Procedures";
      }
      leaf recovery-time {
        type uint32;
        units seconds;
        description
          "RSVP state recovery time";
      }
```

```
    }

    grouping authentication-extended-config {
      description
        "Configuration parameters relating to RSVP
        authentication";
      leaf lifetime {
        type uint32 {
          range "30..86400";
        }
        units seconds;
        description
          "Life time for each security association";
        reference
          "RFC 2747: RSVP Cryptographic
          Authentication";
      }
      leaf window-size {
        type uint32 {
          range "1..64";
        }
        description
          "Window-size to limit number of out-of-order
           messages.";
        reference
          "RFC 2747: RSVP Cryptographic
           Authentication";
      }
      leaf challenge {
        type empty;
        description
          "Enable challenge messages.";
        reference
          "RFC 2747: RSVP Cryptographic
           Authentication";
      }
      leaf retransmits {
        type uint32 {
          range "1..10000";
        }
        description
          "Number of retransmits when messages are
           dropped.";
        reference
          "RFC 2747: RSVP Cryptographic
           Authentication";
      }
      leaf key-chain {
```

```
        type key-chain:key-chain-ref;
        description
          "Key chain name to authenticate RSVP
           signaling messages.";
        reference
          "RFC 2747: RSVP Cryptographic
           Authentication";
      }
    }

    grouping hellos-extended-config {
      description
        "Configuration parameters relating to RSVP
         hellos";
      leaf interface-based {
        type empty;
        description
          "Enable interface-based Hello adjacency if present.";
      }
      leaf hello-interval {
        type uint32;
        units milliseconds;
        description
          "Configure interval between successive Hello
           messages in milliseconds.";
        reference
          "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 signaling-parameters-extended-config {
```

```
      description
        "Configuration parameters relating to RSVP
         signaling";
      leaf refresh-interval {
        type uint32;
        description
          "Set interval between successive refreshes";
      }
      leaf refresh-misses {
        type uint32;
        description
          "Set max number of consecutive missed
          messages for state expiry";
      }
      leaf checksum {
        type boolean;
        description
          "Enable RSVP message checksum computation";
      }
      leaf patherr-state-removal {
        type empty;
        description
          "State-Removal flag in Path Error message
          if present.";
      }
    }

    grouping refresh-reduction-extended-config {
      description
        "Configuration parameters relating to RSVP
        refresh reduction";

      leaf bundle-message-max-size {
        type uint32 {
          range "512..65000";
        }
        description
          "Configure maximum size (bytes) of a
          single RSVP Bundle message.";
      }
      leaf reliable-ack-hold-time {
        type uint32;
        units milliseconds;
        description
          "Configure hold time in milliseconds for
          sending RSVP ACK message(s).";
      }
      leaf reliable-ack-max-size {
```

```
        type uint32;
        description
          "Configure max size of a single RSVP ACK
          message.";
      }
      leaf reliable-retransmit-time {
        type uint32;
        units milliseconds;
        description
          "Configure min delay in milliseconds to
          wait for an ACK before a retransmit.";
      }
      leaf reliable-srefresh {
        type empty;
        description
          "Configure use of reliable messaging for
          summary refresh if present.";
      }
      leaf summary-max-size {
        type uint32 {
          range "20..65000";
        }
        description
          "Configure max size (bytes) of a single
          RSVP summary refresh message.";
      }
    }

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

      leaf in-dropped {
```

```
        type yang:counter64;
        description
          "In packet drop count";
      }

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

      leaf in-error {
        type yang:counter64;
        description
          "In packet rx error count";
      }
    }

    grouping protocol-extended-state {
      description "RSVP protocol statistics.";
    }

    grouping errors-extended-state {
      description
        "Error statistics.";
    }

    grouping extended-state {
      description "RSVP statistic attributes.";
      uses packets-extended-state;
      uses protocol-extended-state;
      uses errors-extended-state;
    }

    /**
     * RSVP extensions augmentations
     */

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

    /* RSVP statistics augmentation */
```

```
    augment "/rt:routing/rt:control-plane-protocols/" +
            "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" +
            "rsvp:statistics/rsvp:state/rsvp:packets" {
      description
        "RSVP packet stats extensions";
      uses packets-extended-state;
    }
    augment "/rt:routing/rt:control-plane-protocols/" +
            "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" +
            "rsvp:statistics/rsvp:state/rsvp:messages" {
      description
        "RSVP protocol message stats extensions";
      uses protocol-extended-state;
    }
    augment "/rt:routing/rt:control-plane-protocols/" +
            "rt:control-plane-protocol/rsvp:rsvp/rsvp:globals/" +
            "rsvp:statistics/rsvp:state/rsvp:errors" {
      description
        "RSVP errors stats extensions";
      uses errors-extended-state;
    }

    /**
     * RSVP all interfaces extensions
     */

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

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

    /* RSVP hellos extension */
    augment "/rt:routing/rt:control-plane-protocols/"
        + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/"
        + "rsvp:hellos" {
```

```
      description
        "RSVP hello all interfaces configuration extensions";
      uses hellos-extended-config;
    }

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

    /**
     * RSVP interface extensions
     */

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

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

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

    /* RSVP authentication extension */
    augment "/rt:routing/rt:control-plane-protocols/"
        + "rt:control-plane-protocol/rsvp:rsvp/rsvp:interfaces/" +
```

```
        "rsvp:interface/rsvp:authentication" {
      description
        "RSVP authentication interface configuration extensions";
      uses authentication-extended-config;
    }
  }
  <CODE ENDS>
```

Figure 5: RSVP extended YANG module

3. IANA Considerations

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

```
   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-extended
   XML: N/A, the requested URI is an XML namespace.
```

This document registers two YANG modules in the YANG Module Names registry [RFC6020].

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

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

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF

users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the

default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

5. Acknowledgement

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

6. Contributors

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones
Brocade

Email: raqib@Brocade.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

7. Normative References

[I-D.ietf-teas-yang-rsvp-te] Beeram, V., Saad, T., Gandhi, R., Liu, X., Bryskin, I., and H. Shah, "A YANG Data Model for RSVP-TE", draft-ietf-teas-yang-rsvp-te-04 (work in progress), October 2018.

[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>.

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <https://www.rfc-editor.org/info/rfc2205>.

[RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, DOI 10.17487/RFC2747, January 2000, <https://www.rfc-editor.org/info/rfc2747>.

[RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC 2961, DOI 10.17487/RFC2961, April 2001, <https://www.rfc-editor.org/info/rfc2961>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <https://www.rfc-editor.org/info/rfc3473>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5063] Satyanarayana, A., Ed. and R. Rahman, Ed., "Extensions to GMPLS Resource Reservation Protocol (RSVP) Graceful Restart", RFC 5063, DOI 10.17487/RFC5063, October 2007, <https://www.rfc-editor.org/info/rfc5063>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

Authors' Addresses

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Tarek Saad (editor)
Cisco Systems, Inc.

Email: tsaad@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.

Email: rgandhi@cisco.com

Xufeng Liu
Jabil

Email: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei Technologies

Email: Igor.Bryskin@huawei.com

Himanshu Shah
Ciena

Email: hshah@ciena.com

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 11, 2019

T. Saad
Juniper Networks
R. Gandhi
Cisco Systems Inc
X. Liu
Volta Networks
V. Beeram
Juniper Networks
I. Bryskin
Huawei Technologies
April 09, 2019

A YANG Data Model for Traffic Engineering Tunnels and Interfaces
draft-ietf-teas-yang-te-21

Abstract

This document defines a YANG data model for the configuration and management of Traffic Engineering (TE) interfaces, tunnels and Label Switched Paths (LSPs). The model is divided into YANG modules that classify data into generic, device-specific, technology agnostic, and technology-specific elements.

This model covers data for configuration, operational state, remote procedural calls, and event notifications.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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

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

Table of Contents

1. Introduction

YANG [RFC6020] and [RFC7950] is a data modeling language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG has proved relevant beyond its initial confines, as bindings to other interfaces (e.g. RESTCONF [RFC8040]) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document describes YANG data model for TE Tunnels, Label Switched Paths (LSPs) and TE interfaces and covers data applicable to generic or device-independent, device-specific, and Multiprotocol Label Switching (MPLS) technology specific.

The document describes a high-level relationship between the modules defined in this document, as well as other external protocol YANG modules. The TE generic YANG data model does not include any data specific to a signaling protocol. It is expected other data plane technology model(s) will augment the TE generic YANG data model.

Also, it is expected other YANG module(s) that model TE signaling protocols, such as RSVP-TE ([RFC3209], [RFC3473]), or Segment-Routing TE (SR-TE) will augment the TE generic YANG module.

1.1. Terminology

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

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

1.2. Prefixes in Data Node Names

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

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
rt-types	ietf-routing-types	[RFC8294]
te	ietf-te	this document
te-dev	ietf-te-device	this document
te-types	ietf-te-types	[I-D.ietf-teas-yang-te-types]
te-mpls-types	ietf-te-mpls-types	[I-D.ietf-teas-yang-te-types]

Table 1: Prefixes and corresponding YANG modules

1.3. TE Technology Models

This document describes the TE generic YANG data model that is independent of any dataplane technology. One of the design objectives is to allow specific data plane technology models to reuse the TE generic data model and possibly augment it with technology specific data.

The elements of the TE generic YANG data model, including TE tunnels, LSPs, and interfaces have leaf(s) that identify the technology layer where they reside. For example, the LSP encoding type can identify the technology associated with a TE tunnel or LSP.

Also, the TE generic YANG data model does not cover signaling protocol data. This is expected to be covered by augmentations defined in other document(s).

1.4. State Data Organization

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

2. Model Overview

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

2.1. Module(s) Relationship

The TE generic YANG data model defined in "ietf-te.yang" covers the building blocks that are device independent and agnostic of any specific technology or control plane instances. The TE device model defined in "ietf-te-device.yang" augments the TE generic YANG data model and covers data that is specific to a device - for example, attributes of TE interfaces, or TE timers that are local to a TE node.

The TE data model for specific instances of data plane technology exist in a separate YANG module(s) that augment the TE generic YANG data model. For example, the MPLS-TE module "ietf-te-mpls.yang" is defined in another document and augments the TE generic model as shown in Figure 1.

The TE data model for specific instances of signaling protocol are outside the scope of this document and are defined in other documents. For example, the RSVP-TE YANG model augmentation of the TE model is covered in [I-D.ietf-teas-yang-rsvp].

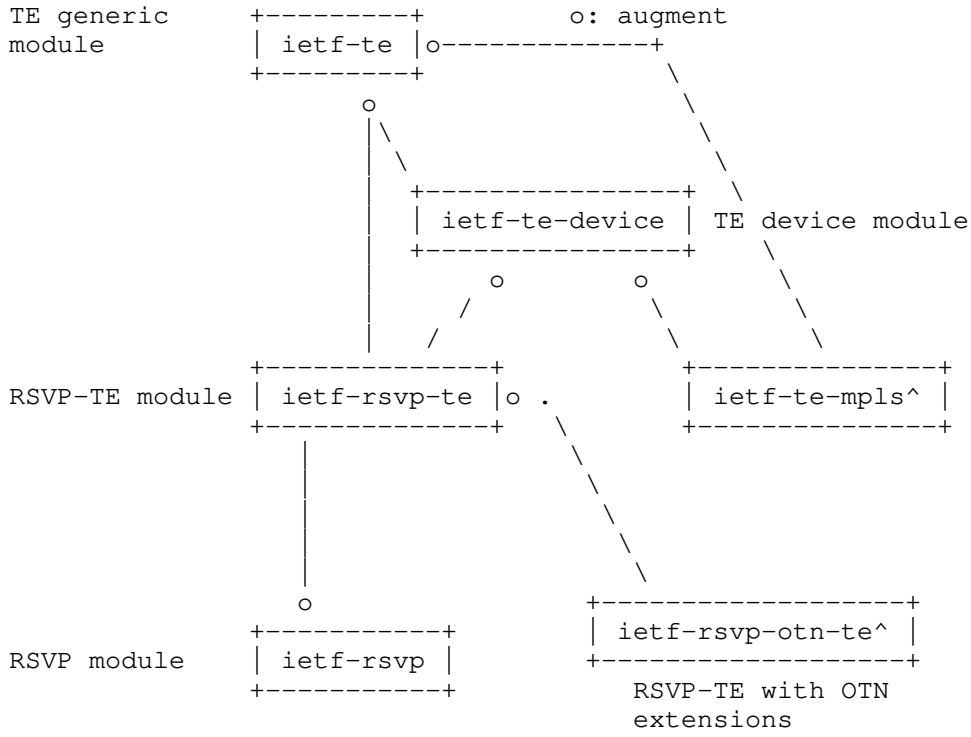
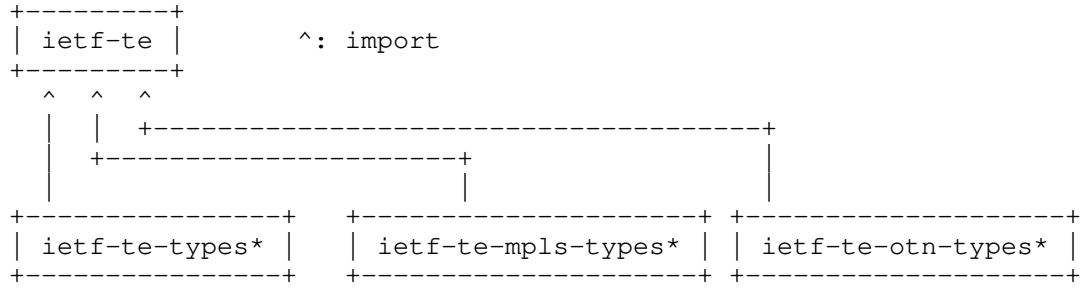


Figure 1: Relationship of TE module(s) with other signaling protocol modules



*: not in this document, shown for illustration only

Figure 2: Relationship between generic and technology specific TE types modules

2.2. Design Considerations

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

- reusable TE data types that are data plane independent are grouped in the TE generic types module "ietf-te-types.yang" defined in [I-D.ietf-teas-yang-te-types]

- reusable TE data types that are data plane specific are defined in a data plane type module, e.g. "ietf-te-packet-types.yang" as defined in [I-D.ietf-teas-yang-te-types]. Other data plane types are expected to be defined in separate module(s) as shown in Figure 2

- The TE generic YANG data model "ietf-te" contains device independent data and can be used to model data off a device (e.g. on a controller). The device-specific TE data is defined in module "ietf-te-device" as shown in Figure 1.

- In general, minimal elements in the model are designated as "mandatory" to allow freedom to vendors to adapt the data model to their specific product implementation.

- This model declares a number of TE functions as features that can be optionally supported.

2.3. Model Tree Diagram

Figure 3 shows the tree diagram of the TE YANG model defined in modules: ietf-te.yang, and ietf-te-device.yang.

```
module: ietf-te
  +--rw te!
     +--rw globals
     |  +--rw named-admin-groups
     |  |  +--rw named-admin-group* [name]
     |  |     +--rw name            string
     |  |     +--rw bit-position?   uint32
     |  +--rw named-srlgs
     |  |  +--rw named-srlg* [name] {te-types:named-srlg-groups}?
     |  |     +--rw name     string
     |  |     +--rw group?   te-types:srlg
     |  |     +--rw cost?    uint32
     |  +--rw named-path-constraints
     |  |  +--rw named-path-constraint* [name]
     |  |           {te-types:named-path-constraints}?
     |  |     +--rw name                              string
```

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

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

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

```
      |   |               |  +--rw (technology)?
      |   |               |     +--:(generic)
      |   |               |        +--rw generic?   int32
      |   |              +--rw range-bitmap?   yang:hex-string
      |   +--rw te-dev:lsp-install-interval?        uint32
      |   +--rw te-dev:lsp-cleanup-interval?        uint32
      |   +--rw te-dev:lsp-invalidation-interval?   uint32
      +--rw tunnels
      |   +--rw tunnel* [name]
      |   |   +--ro operational-state?                  identityref
      |   |   +--rw name                                string
      |   |   +--rw identifier?                         uint16
      |   |   +--rw description?                        string
      |   |   +--rw encoding?                           identityref
      |   |   +--rw switching-type?                     identityref
      |   |   +--rw provisioning-state?                 identityref
      |   |   +--rw preference?                         uint8
      |   |   +--rw reoptimize-timer?                   uint16
      |   |   +--rw source?
      |   |   |       te-types:te-node-id
      |   |   +--rw destination?
      |   |   |       te-types:te-node-id
      |   |   +--rw src-tp-id?                          yang:hex-string
      |   |   +--rw dst-tp-id?                          yang:hex-string
      |   |   +--rw bidirectional?                      boolean
      |   |   +--rw association-objects
      |   |   |  +--rw association-object* [type ID source global-source]
      |   |   |  |  +--rw type             identityref
      |   |   |  |  +--rw ID               uint16
      |   |   |  |  +--rw source           te-types:te-node-id
      |   |   |  |  +--rw global-source    te-types:te-node-id
      |   |   |  +--rw association-object-extended*
      |   |   |  |         [type ID source global-source extended-ID]
      |   |   |  |    +--rw type             identityref
      |   |   |  |    +--rw ID               uint16
      |   |   |  |    +--rw source           te-types:te-node-id
      |   |   |  |    +--rw global-source    te-types:te-node-id
      |   |   |  |    +--rw extended-ID      yang:hex-string
      |   |   +--rw protection
      |   |   |  +--rw enable?                          boolean
      |   |   |  +--rw protection-type?                 identityref
      |   |   |  +--rw protection-reversion-disable?    boolean
      |   |   |  +--rw hold-off-time?                   uint32
      |   |   |  +--rw wait-to-revert?                  uint16
      |   |   |  +--rw aps-signal-id?                   uint8
      |   |   +--rw restoration
      |   |   |  +--rw enable?                          boolean
      |   |   |  +--rw restoration-type?                identityref
```

```
|  |  |  +--rw restoration-scheme?              identityref
|  |  |  +--rw restoration-reversion-disable?   boolean
|  |  |  +--rw hold-off-time?                   uint32
|  |  |  +--rw wait-to-restore?                 uint16
|  |  |  +--rw wait-to-revert?                  uint16
|  |  +--rw te-topology-identifier
|  |  |  +--rw provider-id?   te-global-id
|  |  |  +--rw client-id?     te-global-id
|  |  |  +--rw topology-id?   te-topology-id
|  |  +--rw te-bandwidth
|  |  |  +--rw (technology)?
|  |  |     +--:(generic)
|  |  |        +--rw generic?   te-bandwidth
|  |  +--rw link-protection?                    identityref
|  |  +--rw setup-priority?                     uint8
|  |  +--rw hold-priority?                      uint8
|  |  +--rw signaling-type?                     identityref
|  |  +--rw dependency-tunnels
|  |  |  +--rw dependency-tunnel* [name]
|  |  |     +--rw name
|  |  |     |       -> ../../../../../tunnels/tunnel/name
|  |  |     +--rw encoding?         identityref
|  |  |     +--rw switching-type?   identityref
|  |  +--rw hierarchical-link
|  |  |  +--rw local-te-node-id?         te-types:te-node-id
|  |  |  +--rw local-te-link-tp-id?      te-types:te-tp-id
|  |  |  +--rw remote-te-node-id?        te-types:te-node-id
|  |  |  +--rw te-topology-identifier
|  |  |     +--rw provider-id?   te-global-id
|  |  |     +--rw client-id?     te-global-id
|  |  |     +--rw topology-id?   te-topology-id
|  |  +--rw p2p-primary-paths
|  |  |  +--rw p2p-primary-path* [name]
|  |  |     +--rw name                                string
|  |  |     +--rw path-setup-protocol?                identityref
|  |  |     +--rw path-computation-method?            identityref
|  |  |     +--rw path-computation-server?
|  |  |     |       inet:ip-address
|  |  |     +--rw compute-only?                       empty
|  |  |     +--rw use-path-computation?               boolean
|  |  |     +--rw lockdown?                           empty
|  |  |     +--ro path-scope?                         identityref
|  |  |     +--rw optimizations
|  |  |     |  +--rw (algorithm)?
|  |  |     |     +--:(metric) {path-optimization-metric}?
|  |  |     |     |  +--rw optimization-metric* [metric-type]
|  |  |     |     |     |  +--rw metric-type
|  |  |     |     |     |  |       identityref
```

```
      |  |  |  |  |  |  +--rw weight?
      |  |  |  |  |  |  |       uint8
      |  |  |  |  |  |  +--rw explicit-route-exclude-objects
      |  |  |  |  |  |  |  +--rw route-object-exclude-object*
      |  |  |  |  |  |  |          [index]
      |  |  |  |  |  |  |     +--rw index
      |  |  |  |  |  |  |     |       uint32
      |  |  |  |  |  |  |     +--rw (type)?
      |  |  |  |  |  |  |        +--:(numbered-node-hop)
      |  |  |  |  |  |  |        |  +--rw numbered-node-hop
      |  |  |  |  |  |  |        |     +--rw node-id
      |  |  |  |  |  |  |        |     |       te-node-id
      |  |  |  |  |  |  |        |     +--rw hop-type?
      |  |  |  |  |  |  |        |             te-hop-type
      |  |  |  |  |  |  |        +--:(numbered-link-hop)
      |  |  |  |  |  |  |        |  +--rw numbered-link-hop
      |  |  |  |  |  |  |        |     +--rw link-tp-id
      |  |  |  |  |  |  |        |     |       te-tp-id
      |  |  |  |  |  |  |        |     +--rw hop-type?
      |  |  |  |  |  |  |        |     |       te-hop-type
      |  |  |  |  |  |  |        |     +--rw direction?
      |  |  |  |  |  |  |        |             te-link-direction
      |  |  |  |  |  |  |        +--:(unnumbered-link-hop)
      |  |  |  |  |  |  |        |  +--rw unnumbered-link-hop
      |  |  |  |  |  |  |        |     +--rw link-tp-id
      |  |  |  |  |  |  |        |     |       te-tp-id
      |  |  |  |  |  |  |        |     +--rw node-id
      |  |  |  |  |  |  |        |     |       te-node-id
      |  |  |  |  |  |  |        |     +--rw hop-type?
      |  |  |  |  |  |  |        |     |       te-hop-type
      |  |  |  |  |  |  |        |     +--rw direction?
      |  |  |  |  |  |  |        |             te-link-direction
      |  |  |  |  |  |  |        +--:(as-number)
      |  |  |  |  |  |  |        |  +--rw as-number-hop
      |  |  |  |  |  |  |        |     +--rw as-number
      |  |  |  |  |  |  |        |     |       inet:as-number
      |  |  |  |  |  |  |        |     +--rw hop-type?
      |  |  |  |  |  |  |        |             te-hop-type
      |  |  |  |  |  |  |        +--:(label)
      |  |  |  |  |  |  |        |  +--rw label-hop
      |  |  |  |  |  |  |        |     +--rw te-label
      |  |  |  |  |  |  |        |        +--rw (technology)?
      |  |  |  |  |  |  |        |        |  +--:(generic)
      |  |  |  |  |  |  |        |        |     +--rw generic?
      |  |  |  |  |  |  |        |       rt-types:generalized-label
      |  |  |  |  |  |  |        |        +--rw direction?
      |  |  |  |  |  |  |        |       te-label-direction
      |  |  |  |  |  |  |        +--:(srlg)
```

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

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

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

```
      |  |  |  |  |  |     +--rw (technology)?
      |  |  |  |  |  |     |  +--:(generic)
      |  |  |  |  |  |     |     +--rw generic?
      |  |  |  |  |  |     |             rt-types:generalized-label
      |  |  |  |  |  |     +--rw direction?
      |  |  |  |  |  |             te-label-direction
      |  |  |  |  |  +--rw label-end
      |  |  |  |  |  |  +--rw te-label
      |  |  |  |  |  |     +--rw (technology)?
      |  |  |  |  |  |     |  +--:(generic)
      |  |  |  |  |  |     |     +--rw generic?
      |  |  |  |  |  |     |             rt-types:generalized-label
      |  |  |  |  |  |     +--rw direction?
      |  |  |  |  |  |             te-label-direction
      |  |  |  |  |  +--rw label-step
      |  |  |  |  |  |  +--rw (technology)?
      |  |  |  |  |  |     +--:(generic)
      |  |  |  |  |  |        +--rw generic?   int32
      |  |  |  |  |  +--rw range-bitmap?   yang:hex-string
      |  |  |  |  +--ro computed-paths-properties
      |  |  |  |  |  +--ro computed-path-properties* [k-index]
      |  |  |  |  |     +--ro k-index            uint8
      |  |  |  |  |     +--ro path-properties
      |  |  |  |  |        +--ro path-metric* [metric-type]
      |  |  |  |  |        |  +--ro metric-type           identityref
      |  |  |  |  |        |  +--ro accumulative-value?   uint64
      |  |  |  |  |        +--ro path-affinities-values
      |  |  |  |  |        |  +--ro path-affinities-value* [usage]
      |  |  |  |  |        |     +--ro usage    identityref
      |  |  |  |  |        |     +--ro value?   admin-groups
      |  |  |  |  |        +--ro path-affinity-names
      |  |  |  |  |        |  +--ro path-affinity-name* [usage]
      |  |  |  |  |        |     +--ro usage            identityref
      |  |  |  |  |        |     +--ro affinity-name* [name]
      |  |  |  |  |        |        +--ro name    string
      |  |  |  |  |        +--ro path-srlgs-lists
      |  |  |  |  |        |  +--ro path-srlgs-list* [usage]
      |  |  |  |  |        |     +--ro usage     identityref
      |  |  |  |  |        |     +--ro values*   srlg
      |  |  |  |  |        +--ro path-srlgs-names
      |  |  |  |  |        |  +--ro path-srlgs-name* [usage]
      |  |  |  |  |        |     +--ro usage    identityref
      |  |  |  |  |        |     +--ro names*   string
      |  |  |  |  |        +--ro path-route-objects
      |  |  |  |  |        |  +--ro path-computed-route-object* [index]
      |  |  |  |  |        |     +--ro index
      |  |  |  |  |        |     |       uint32
      |  |  |  |  |        |     +--ro (type)?
```

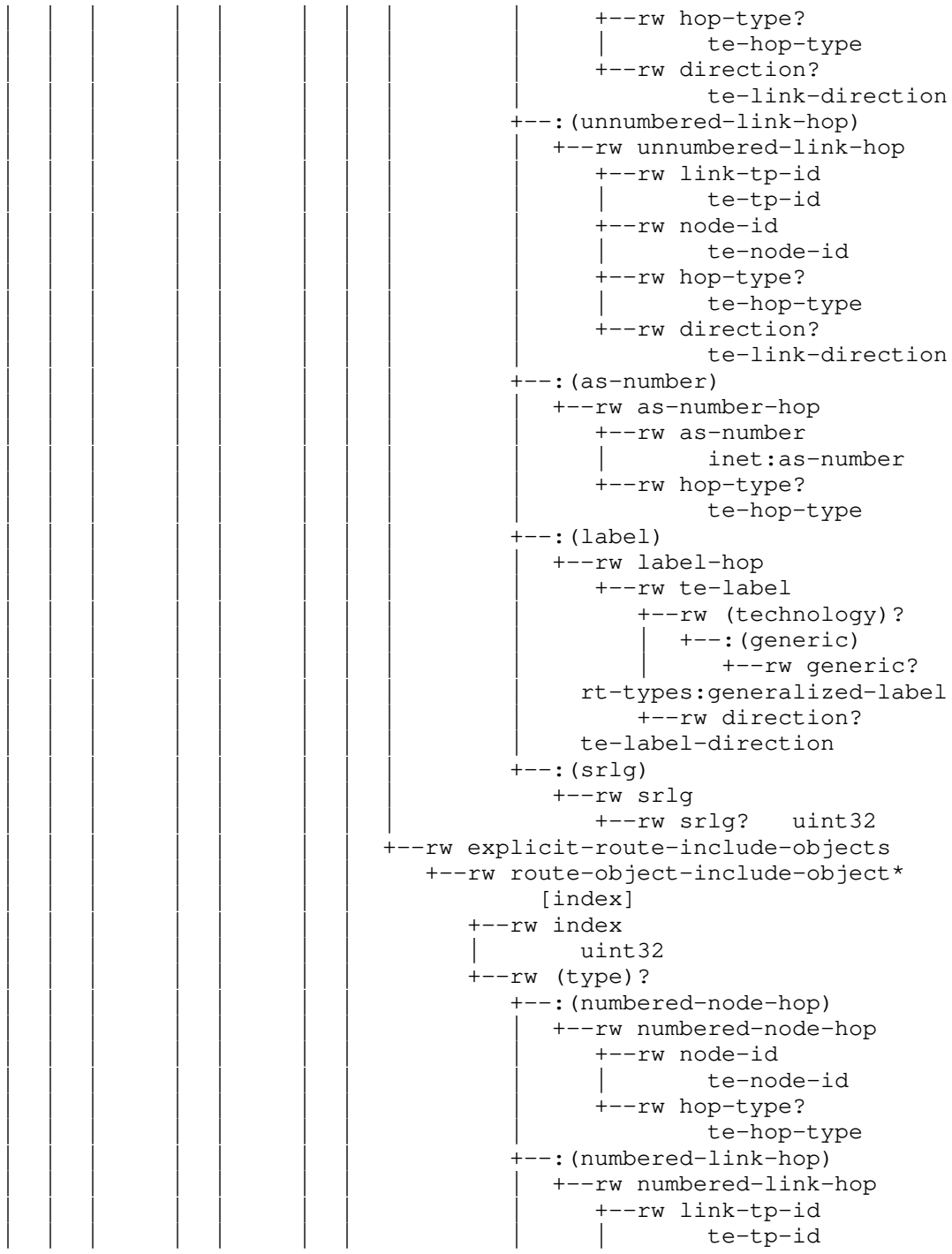
```
      |  |  |      |         |        +--:(numbered-node-hop)
      |  |  |      |         |        |  +--ro numbered-node-hop
      |  |  |      |         |        |     +--ro node-id     te-node-id
      |  |  |      |         |        |     +--ro hop-type?   te-hop-type
      |  |  |      |         |        +--:(numbered-link-hop)
      |  |  |      |         |        |  +--ro numbered-link-hop
      |  |  |      |         |        |     +--ro link-tp-id    te-tp-id
      |  |  |      |         |        |     +--ro hop-type?
      |  |  |      |         |        |     |       te-hop-type
      |  |  |      |         |        |     +--ro direction?
      |  |  |      |         |        |             te-link-direction
      |  |  |      |         |        +--:(unnumbered-link-hop)
      |  |  |      |         |        |  +--ro unnumbered-link-hop
      |  |  |      |         |        |     +--ro link-tp-id    te-tp-id
      |  |  |      |         |        |     +--ro node-id
      |  |  |      |         |        |     |       te-node-id
      |  |  |      |         |        |     +--ro hop-type?
      |  |  |      |         |        |     |       te-hop-type
      |  |  |      |         |        |     +--ro direction?
      |  |  |      |         |        |             te-link-direction
      |  |  |      |         |        +--:(as-number)
      |  |  |      |         |        |  +--ro as-number-hop
      |  |  |      |         |        |     +--ro as-number
      |  |  |      |         |        |     |       inet:as-number
      |  |  |      |         |        |     +--ro hop-type?
      |  |  |      |         |        |             te-hop-type
      |  |  |      |         |        +--:(label)
      |  |  |      |         |           +--ro label-hop
      |  |  |      |         |              +--ro te-label
      |  |  |      |         |                 +--ro (technology)?
      |  |  |      |         |                 |  +--:(generic)
      |  |  |      |         |                 |     +--ro generic?
      |  |  |      |         |               rt-types:generalized-label
      |  |  |      |         |                +--ro direction?
      |  |  |      |         |                        te-label-direction
      |  |  |      |        +--ro shared-resources-tunnels
      |  |  |      |           +--ro lsp-shared-resources-tunnel*
      |  |  |      |                   tunnel-ref
      |  |  |     +--ro computed-path-error-infos
      |  |  |     |  +--ro computed-path-error-info* []
      |  |  |     |     +--ro error-description?   string
      |  |  |     |     +--ro error-timestamp?     yang:date-and-time
      |  |  |     |     +--ro error-reason?        identityref
      |  |  |     +--ro lsp-provisioning-error-infos
      |  |  |     |  +--ro lsp-provisioning-error-info* []
      |  |  |     |     +--ro error-description?   string
      |  |  |     |     +--ro error-timestamp?     yang:date-and-time
      |  |  |     |     +--ro error-node-id?       te-types:te-node-id
```

```
      |  |  |  |     +--ro error-link-id?       te-types:te-tp-id
      |  |  |  |     +--ro lsp-id?              uint16
      |  |  |  |   +--ro lsps
      |  |  |  |     +--ro lsp* [lsp-id]
      |  |  |  |        +--ro lsp-provisioning-error-infos
      |  |  |  |        |  +--ro lsp-provisioning-error-info* []
      |  |  |  |        |     +--ro error-description?   string
      |  |  |  |        |     +--ro error-timestamp?
      |  |  |  |        |     |       yang:date-and-time
      |  |  |  |        |     +--ro error-node-id?
      |  |  |  |        |     |       te-types:te-node-id
      |  |  |  |        |     +--ro error-link-id?
      |  |  |  |        |             te-types:te-tp-id
      |  |  |  |        +--ro source?
      |  |  |  |        |       te-types:te-node-id
      |  |  |  |        +--ro destination?
      |  |  |  |        |       te-types:te-node-id
      |  |  |  |        +--ro tunnel-id?
      |  |  |  |        |       uint16
      |  |  |  |        +--ro lsp-id
      |  |  |  |        |       uint16
      |  |  |  |        +--ro extended-tunnel-id?
      |  |  |  |        |       yang:dotted-quad
      |  |  |  |        +--ro operational-state?
      |  |  |  |        |       identityref
      |  |  |  |        +--ro path-setup-protocol?
      |  |  |  |        |       identityref
      |  |  |  |        +--ro origin-type?
      |  |  |  |        |       enumeration
      |  |  |  |        +--ro lsp-resource-status?
      |  |  |  |        |       enumeration
      |  |  |  |        +--ro lockout-of-normal?
      |  |  |  |        |       boolean
      |  |  |  |        +--ro freeze?
      |  |  |  |        |       boolean
      |  |  |  |        +--ro lsp-protection-role?
      |  |  |  |        |       enumeration
      |  |  |  |        +--ro lsp-protection-state?
      |  |  |  |        |       identityref
      |  |  |  |        +--ro protection-group-ingress-node-id?
      |  |  |  |        |       te-types:te-node-id
      |  |  |  |        +--ro protection-group-egress-node-id?
      |  |  |  |        |       te-types:te-node-id
      |  |  |  |        +--ro lsp-shared-resources-tunnel?
      |  |  |  |        |       tunnel-ref
      |  |  |  |        +--ro lsp-record-route-information
      |  |  |  |        |  +--ro lsp-record-route-information* [index]
      |  |  |  |        |     +--ro index                        uint32
```

```
      |  |  |  |  |  |     +--ro (type)?
      |  |  |  |  |  |        +--:(numbered-node-hop)
      |  |  |  |  |  |        |  +--ro numbered-node-hop
      |  |  |  |  |  |        |     +--ro node-id    te-node-id
      |  |  |  |  |  |        |     +--ro flags*
      |  |  |  |  |  |        |              path-attribute-flags
      |  |  |  |  |  |        +--:(numbered-link-hop)
      |  |  |  |  |  |        |  +--ro numbered-link-hop
      |  |  |  |  |  |        |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |        |     +--ro flags*
      |  |  |  |  |  |        |              path-attribute-flags
      |  |  |  |  |  |        +--:(unnumbered-link-hop)
      |  |  |  |  |  |        |  +--ro unnumbered-link-hop
      |  |  |  |  |  |        |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |        |     +--ro node-id?      te-node-id
      |  |  |  |  |  |        |     +--ro flags*
      |  |  |  |  |  |        |              path-attribute-flags
      |  |  |  |  |  |        +--:(label)
      |  |  |  |  |  |           +--ro label-hop
      |  |  |  |  |  |              +--ro te-label
      |  |  |  |  |  |              |  +--ro (technology)?
      |  |  |  |  |  |              |  |  +--:(generic)
      |  |  |  |  |  |              |  |     +--ro generic?
      |  |  |  |  |  |              |  |     rt-types:generalized-label
      |  |  |  |  |  |              |  +--ro direction?
      |  |  |  |  |  |              |           te-label-direction
      |  |  |  |  |  |              +--ro flags*
      |  |  |  |  |  |                       path-attribute-flags
      |  |  |  |  |  +--ro path-properties
      |  |  |  |  |     +--ro path-metric* [metric-type]
      |  |  |  |  |     |  +--ro metric-type           identityref
      |  |  |  |  |     |  +--ro accumulative-value?   uint64
      |  |  |  |  |     +--ro path-affinities-values
      |  |  |  |  |     |  +--ro path-affinities-value* [usage]
      |  |  |  |  |     |     +--ro usage    identityref
      |  |  |  |  |     |     +--ro value?   admin-groups
      |  |  |  |  |     +--ro path-affinity-names
      |  |  |  |  |     |  +--ro path-affinity-name* [usage]
      |  |  |  |  |     |     +--ro usage            identityref
      |  |  |  |  |     |     +--ro affinity-name* [name]
      |  |  |  |  |     |        +--ro name    string
      |  |  |  |  |     +--ro path-srlgs-lists
      |  |  |  |  |     |  +--ro path-srlgs-list* [usage]
      |  |  |  |  |     |     +--ro usage     identityref
      |  |  |  |  |     |     +--ro values*   srlg
      |  |  |  |  |     +--ro path-srlgs-names
      |  |  |  |  |     |  +--ro path-srlgs-name* [usage]
      |  |  |  |  |     |     +--ro usage    identityref
```

```
      |  |  |  |  |  |     +--ro names*   string
      |  |  |  |  |  |   +--ro path-route-objects
      |  |  |  |  |  |   |  +--ro path-computed-route-object* [index]
      |  |  |  |  |  |   |     +--ro index
      |  |  |  |  |  |   |     |       uint32
      |  |  |  |  |  |   |     +--ro (type)?
      |  |  |  |  |  |   |        +--:(numbered-node-hop)
      |  |  |  |  |  |   |        |  +--ro numbered-node-hop
      |  |  |  |  |  |   |        |     +--ro node-id     te-node-id
      |  |  |  |  |  |   |        |     +--ro hop-type?   te-hop-type
      |  |  |  |  |  |   |        +--:(numbered-link-hop)
      |  |  |  |  |  |   |        |  +--ro numbered-link-hop
      |  |  |  |  |  |   |        |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |   |        |     +--ro hop-type?
      |  |  |  |  |  |   |        |     |       te-hop-type
      |  |  |  |  |  |   |        |     +--ro direction?
      |  |  |  |  |  |   |        |             te-link-direction
      |  |  |  |  |  |   |        +--:(unnumbered-link-hop)
      |  |  |  |  |  |   |        |  +--ro unnumbered-link-hop
      |  |  |  |  |  |   |        |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |   |        |     +--ro node-id
      |  |  |  |  |  |   |        |     |       te-node-id
      |  |  |  |  |  |   |        |     +--ro hop-type?
      |  |  |  |  |  |   |        |     |       te-hop-type
      |  |  |  |  |  |   |        |     +--ro direction?
      |  |  |  |  |  |   |        |             te-link-direction
      |  |  |  |  |  |   |        +--:(as-number)
      |  |  |  |  |  |   |        |  +--ro as-number-hop
      |  |  |  |  |  |   |        |     +--ro as-number
      |  |  |  |  |  |   |        |     |       inet:as-number
      |  |  |  |  |  |   |        |     +--ro hop-type?
      |  |  |  |  |  |   |        |             te-hop-type
      |  |  |  |  |  |   |        +--:(label)
      |  |  |  |  |  |   |           +--ro label-hop
      |  |  |  |  |  |   |              +--ro te-label
      |  |  |  |  |  |   |                 +--ro (technology)?
      |  |  |  |  |  |   |                 |  +--:(generic)
      |  |  |  |  |  |   |                 |     +--ro generic?
      |  |  |  |  |  |   |                 rt-types:generalized-label
      |  |  |  |  |  |   |                  +--ro direction?
      |  |  |  |  |  |   |                          te-label-direction
      |  |  |  |  |  |   +--ro shared-resources-tunnels
      |  |  |  |  |  |      +--ro lsp-shared-resources-tunnel*
      |  |  |  |  |  |             tunnel-ref
      |  |  |  |  |  +--ro te-dev:lsp-timers
      |  |  |  |  |  |  +--ro te-dev:life-time?         uint32
      |  |  |  |  |  |  +--ro te-dev:time-to-install?   uint32
      |  |  |  |  |  |  +--ro te-dev:time-to-destroy?   uint32
```

```
      |  |  |      |     +--ro te-dev:downstream-info
      |  |  |      |     |  +--ro te-dev:nhop?
      |  |  |      |     |  |       inet:ip-address
      |  |  |      |     |  +--ro te-dev:outgoing-interface?
      |  |  |      |     |  |       if:interface-ref
      |  |  |      |     |  +--ro te-dev:neighbor?
      |  |  |      |     |  |       inet:ip-address
      |  |  |      |     |  +--ro te-dev:label?
      |  |  |      |     |          rt-types:generalized-label
      |  |  |      |     +--ro te-dev:upstream-info
      |  |  |      |        +--ro te-dev:phop?       inet:ip-address
      |  |  |      |        +--ro te-dev:neighbor?   inet:ip-address
      |  |  |      |        +--ro te-dev:label?
      |  |  |      |                rt-types:generalized-label
      |  |  |      +--rw p2p-primary-reverse-path
      |  |  |      |  +--rw name?                                string
      |  |  |      |  +--rw path-setup-protocol?                 identityref
      |  |  |      |  +--rw path-computation-method?             identityref
      |  |  |      |  +--rw path-computation-server?
      |  |  |      |  |       inet:ip-address
      |  |  |      |  +--rw compute-only?                        empty
      |  |  |      |  +--rw use-path-computation?                boolean
      |  |  |      |  +--rw lockdown?                            empty
      |  |  |      |  +--ro path-scope?                          identityref
      |  |  |      |  +--rw optimizations
      |  |  |      |  |  +--rw (algorithm)?
      |  |  |      |  |     +--:(metric) {path-optimization-metric}?
      |  |  |      |  |     |  +--rw optimization-metric* [metric-type]
      |  |  |      |  |     |  |  +--rw metric-type
      |  |  |      |  |     |  |  |       identityref
      |  |  |      |  |     |  |  +--rw weight?
      |  |  |      |  |     |  |  |       uint8
      |  |  |      |  |     |  |  +--rw explicit-route-exclude-objects
      |  |  |      |  |     |  |  |  +--rw route-object-exclude-object*
      |  |  |      |  |     |  |  |          [index]
      |  |  |      |  |     |  |  |     +--rw index
      |  |  |      |  |     |  |  |     |       uint32
      |  |  |      |  |     |  |  |     +--rw (type)?
      |  |  |      |  |     |  |  |        +--:(numbered-node-hop)
      |  |  |      |  |     |  |  |        |  +--rw numbered-node-hop
      |  |  |      |  |     |  |  |        |     +--rw node-id
      |  |  |      |  |     |  |  |        |     |       te-node-id
      |  |  |      |  |     |  |  |        |     +--rw hop-type?
      |  |  |      |  |     |  |  |        |             te-hop-type
      |  |  |      |  |     |  |  |        +--:(numbered-link-hop)
      |  |  |      |  |     |  |  |        |  +--rw numbered-link-hop
      |  |  |      |  |     |  |  |        |     +--rw link-tp-id
      |  |  |      |  |     |  |  |        |     |       te-tp-id
```


+--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
      |  |  |       |  +--rw named-path-constraint?           leafref
      |  |  |       |  |       {te-types:named-path-constraints}?
      |  |  |       |  +--rw te-bandwidth
      |  |  |       |  |  +--rw (technology)?
      |  |  |       |  |     +--:(generic)
      |  |  |       |  |        +--rw generic?   te-bandwidth
      |  |  |       |  +--rw link-protection?                 identityref
      |  |  |       |  +--rw setup-priority?                  uint8
      |  |  |       |  +--rw hold-priority?                   uint8
      |  |  |       |  +--rw signaling-type?                  identityref
```

```
      |   |   |       |   +--rw path-metric-bounds
      |   |   |       |   |  +--rw path-metric-bound* [metric-type]
      |   |   |       |   |     +--rw metric-type    identityref
      |   |   |       |   |     +--rw upper-bound?   uint64
      |   |   |       |   +--rw path-affinities-values
      |   |   |       |   |  +--rw path-affinities-value* [usage]
      |   |   |       |   |     +--rw usage    identityref
      |   |   |       |   |     +--rw value?   admin-groups
      |   |   |       |   +--rw path-affinity-names
      |   |   |       |   |  +--rw path-affinity-name* [usage]
      |   |   |       |   |     +--rw usage            identityref
      |   |   |       |   |     +--rw affinity-name* [name]
      |   |   |       |   |        +--rw name    string
      |   |   |       |   +--rw path-srlgs-lists
      |   |   |       |   |  +--rw path-srlgs-list* [usage]
      |   |   |       |   |     +--rw usage     identityref
      |   |   |       |   |     +--rw values*   srlg
      |   |   |       |   +--rw path-srlgs-names
      |   |   |       |   |  +--rw path-srlgs-name* [usage]
      |   |   |       |   |     +--rw usage    identityref
      |   |   |       |   |     +--rw names*   string
      |   |   |       |   +--rw disjointness?
      |   |   |       |   |       te-path-disjointness
      |   |   |       |   +--rw explicit-route-objects-always
      |   |   |       |   |  +--rw route-object-exclude-always* [index]
      |   |   |       |   |  |  +--rw index                          uint32
      |   |   |       |   |  |  +--rw (type)?
      |   |   |       |   |  |     +--:(numbered-node-hop)
      |   |   |       |   |  |     |  +--rw numbered-node-hop
      |   |   |       |   |  |     |     +--rw node-id     te-node-id
      |   |   |       |   |  |     |     +--rw hop-type?   te-hop-type
      |   |   |       |   |  |     +--:(numbered-link-hop)
      |   |   |       |   |  |     |  +--rw numbered-link-hop
      |   |   |       |   |  |     |     +--rw link-tp-id    te-tp-id
      |   |   |       |   |  |     |     +--rw hop-type?     te-hop-type
      |   |   |       |   |  |     |     +--rw direction?
      |   |   |       |   |  |     |             te-link-direction
      |   |   |       |   |  |     +--:(unnumbered-link-hop)
      |   |   |       |   |  |     |  +--rw unnumbered-link-hop
      |   |   |       |   |  |     |     +--rw link-tp-id    te-tp-id
      |   |   |       |   |  |     |     +--rw node-id       te-node-id
      |   |   |       |   |  |     |     +--rw hop-type?     te-hop-type
      |   |   |       |   |  |     |     +--rw direction?
      |   |   |       |   |  |     |             te-link-direction
      |   |   |       |   |  |     +--:(as-number)
      |   |   |       |   |  |     |  +--rw as-number-hop
      |   |   |       |   |  |     |     +--rw as-number    inet:as-number
      |   |   |       |   |  |     |     +--rw hop-type?    te-hop-type
```

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

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

```
      |   |   |       |   |         +--rw label-step
      |   |   |       |   |         |  +--rw (technology)?
      |   |   |       |   |         |     +--:(generic)
      |   |   |       |   |         |        +--rw generic?   int32
      |   |   |       |   |         +--rw range-bitmap?   yang:hex-string
      |   |   |       |  +--ro computed-paths-properties
      |   |   |       |   |  +--ro computed-path-properties* [k-index]
      |   |   |       |   |     +--ro k-index            uint8
      |   |   |       |   |     +--ro path-properties
      |   |   |       |   |        +--ro path-metric* [metric-type]
      |   |   |       |   |        |  +--ro metric-type
      |   |   |       |   |        |  |       identityref
      |   |   |       |   |        |  +--ro accumulative-value?   uint64
      |   |   |       |   |        +--ro path-affinities-values
      |   |   |       |   |        |  +--ro path-affinities-value* [usage]
      |   |   |       |   |        |     +--ro usage    identityref
      |   |   |       |   |        |     +--ro value?   admin-groups
      |   |   |       |   |        +--ro path-affinity-names
      |   |   |       |   |        |  +--ro path-affinity-name* [usage]
      |   |   |       |   |        |     +--ro usage            identityref
      |   |   |       |   |        |     +--ro affinity-name* [name]
      |   |   |       |   |        |        +--ro name    string
      |   |   |       |   |        +--ro path-srlgs-lists
      |   |   |       |   |        |  +--ro path-srlgs-list* [usage]
      |   |   |       |   |        |     +--ro usage     identityref
      |   |   |       |   |        |     +--ro values*   srlg
      |   |   |       |   |        +--ro path-srlgs-names
      |   |   |       |   |        |  +--ro path-srlgs-name* [usage]
      |   |   |       |   |        |     +--ro usage    identityref
      |   |   |       |   |        |     +--ro names*   string
      |   |   |       |   |        +--ro path-route-objects
      |   |   |       |   |        |  +--ro path-computed-route-object*
      |   |   |       |   |        |           [index]
      |   |   |       |   |        |     +--ro index
      |   |   |       |   |        |     |       uint32
      |   |   |       |   |        |     +--ro (type)?
      |   |   |       |   |        |        +--:(numbered-node-hop)
      |   |   |       |   |        |        |  +--ro numbered-node-hop
      |   |   |       |   |        |        |     +--ro node-id
      |   |   |       |   |        |        |     |       te-node-id
      |   |   |       |   |        |        |     +--ro hop-type?
      |   |   |       |   |        |        |              te-hop-type
      |   |   |       |   |        |        +--:(numbered-link-hop)
      |   |   |       |   |        |        |  +--ro numbered-link-hop
      |   |   |       |   |        |        |     +--ro link-tp-id
      |   |   |       |   |        |        |     |       te-tp-id
      |   |   |       |   |        |        |     +--ro hop-type?
      |   |   |       |   |        |        |     |       te-hop-type
```

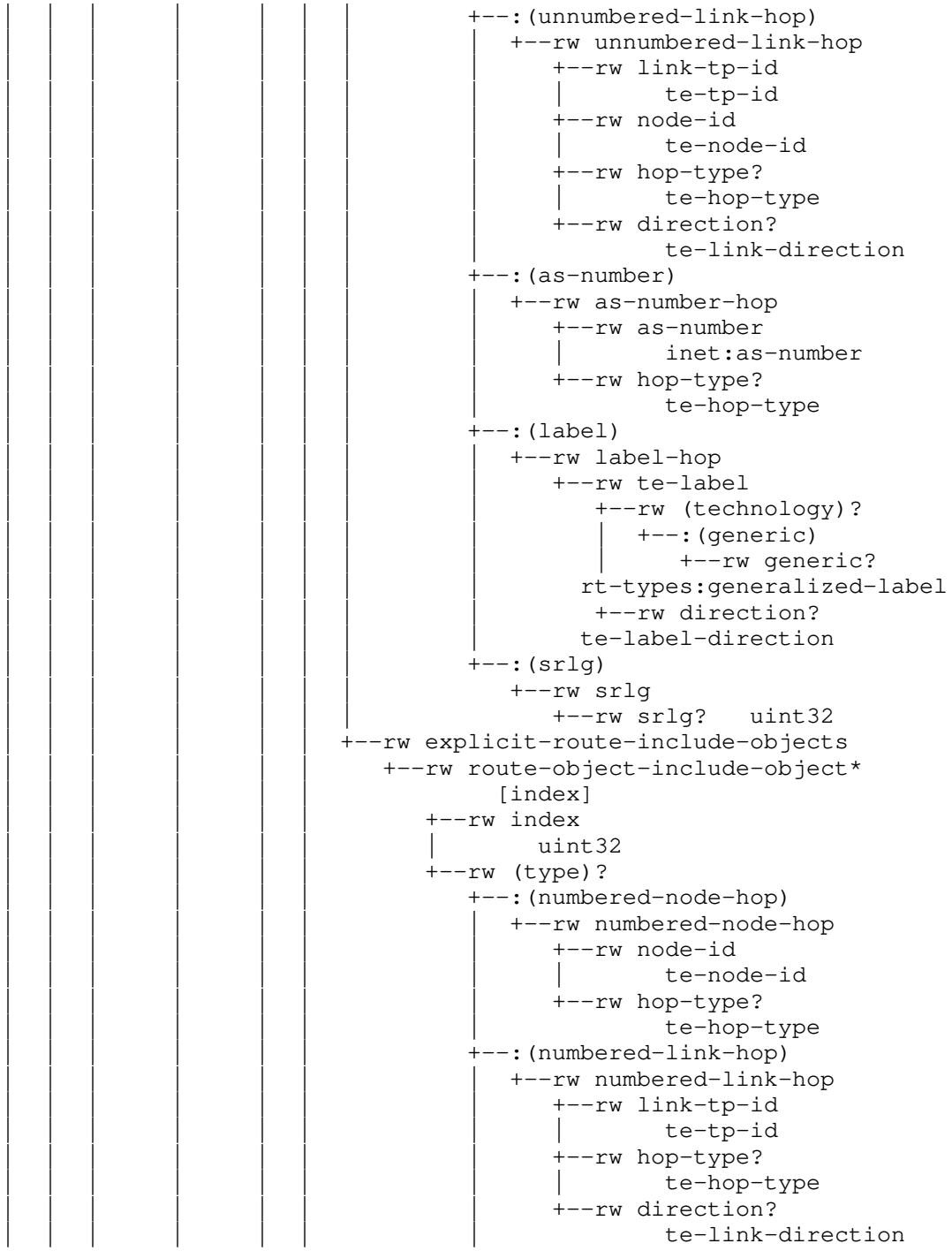
```
      |  |  |  |  |  |  |  |     +--ro direction?
      |  |  |  |  |  |  |  |              te-link-direction
      |  |  |  |  |  |  |  +--:(unnumbered-link-hop)
      |  |  |  |  |  |  |  |  +--ro unnumbered-link-hop
      |  |  |  |  |  |  |  |     +--ro link-tp-id
      |  |  |  |  |  |  |  |     |       te-tp-id
      |  |  |  |  |  |  |  |     +--ro node-id
      |  |  |  |  |  |  |  |     |       te-node-id
      |  |  |  |  |  |  |  |     +--ro hop-type?
      |  |  |  |  |  |  |  |     |       te-hop-type
      |  |  |  |  |  |  |  |     +--ro direction?
      |  |  |  |  |  |  |  |              te-link-direction
      |  |  |  |  |  |  |  +--:(as-number)
      |  |  |  |  |  |  |  |  +--ro as-number-hop
      |  |  |  |  |  |  |  |     +--ro as-number
      |  |  |  |  |  |  |  |     |       inet:as-number
      |  |  |  |  |  |  |  |     +--ro hop-type?
      |  |  |  |  |  |  |  |              te-hop-type
      |  |  |  |  |  |  |  +--:(label)
      |  |  |  |  |  |  |     +--ro label-hop
      |  |  |  |  |  |  |        +--ro te-label
      |  |  |  |  |  |  |           +--ro (technology)?
      |  |  |  |  |  |  |           |  +--:(generic)
      |  |  |  |  |  |  |           |     +--ro generic?
      |  |  |  |  |  |  |            rt-types:generalized-label
      |  |  |  |  |  |  |              +--ro direction?
      |  |  |  |  |  |  |             te-label-direction
      |  |  |  |  |  |  +--ro shared-resources-tunnels
      |  |  |  |  |  |     +--ro lsp-shared-resources-tunnel*
      |  |  |  |  |  |               tunnel-ref
      |  |  |  |  +--ro computed-path-error-infos
      |  |  |  |  |  +--ro computed-path-error-info* []
      |  |  |  |  |     +--ro error-description?   string
      |  |  |  |  |     +--ro error-timestamp?
      |  |  |  |  |     |       yang:date-and-time
      |  |  |  |  |     +--ro error-reason?        identityref
      |  |  |  |  +--ro lsp-provisioning-error-infos
      |  |  |  |  |  +--ro lsp-provisioning-error-info* []
      |  |  |  |  |     +--ro error-description?   string
      |  |  |  |  |     +--ro error-timestamp?
      |  |  |  |  |     |       yang:date-and-time
      |  |  |  |  |     +--ro error-node-id?
      |  |  |  |  |     |       te-types:te-node-id
      |  |  |  |  |     +--ro error-link-id?       te-types:te-tp-id
      |  |  |  |  |     +--ro lsp-id?              uint16
      |  |  |  |  +--ro lsps
      |  |  |  |  |  +--ro lsp* [lsp-id]
      |  |  |  |  |     +--ro lsp-provisioning-error-infos
```

```
      |  |  |      |  |      |  +--ro lsp-provisioning-error-info* []
      |  |  |      |  |      |     +--ro error-description?   string
      |  |  |      |  |      |     +--ro error-timestamp?
      |  |  |      |  |      |     |       yang:date-and-time
      |  |  |      |  |      |     +--ro error-node-id?
      |  |  |      |  |      |     |       te-types:te-node-id
      |  |  |      |  |      |     +--ro error-link-id?
      |  |  |      |  |      |             te-types:te-tp-id
      |  |  |      |  |      +--ro source?
      |  |  |      |  |      |       te-types:te-node-id
      |  |  |      |  |      +--ro destination?
      |  |  |      |  |      |       te-types:te-node-id
      |  |  |      |  |      +--ro tunnel-id?
      |  |  |      |  |      |       uint16
      |  |  |      |  |      +--ro lsp-id
      |  |  |      |  |      |       uint16
      |  |  |      |  |      +--ro extended-tunnel-id?
      |  |  |      |  |      |       yang:dotted-quad
      |  |  |      |  |      +--ro operational-state?
      |  |  |      |  |      |       identityref
      |  |  |      |  |      +--ro path-setup-protocol?
      |  |  |      |  |      |       identityref
      |  |  |      |  |      +--ro origin-type?
      |  |  |      |  |      |       enumeration
      |  |  |      |  |      +--ro lsp-resource-status?
      |  |  |      |  |      |       enumeration
      |  |  |      |  |      +--ro lockout-of-normal?
      |  |  |      |  |      |       boolean
      |  |  |      |  |      +--ro freeze?
      |  |  |      |  |      |       boolean
      |  |  |      |  |      +--ro lsp-protection-role?
      |  |  |      |  |      |       enumeration
      |  |  |      |  |      +--ro lsp-protection-state?
      |  |  |      |  |      |       identityref
      |  |  |      |  |      +--ro protection-group-ingress-node-id?
      |  |  |      |  |      |       te-types:te-node-id
      |  |  |      |  |      +--ro protection-group-egress-node-id?
      |  |  |      |  |      |       te-types:te-node-id
      |  |  |      |  |      +--ro lsp-shared-resources-tunnel?
      |  |  |      |  |      |       tunnel-ref
      |  |  |      |  |      +--ro lsp-record-route-information
      |  |  |      |  |      |  +--ro lsp-record-route-information*
      |  |  |      |  |      |           [index]
      |  |  |      |  |      |     +--ro index
      |  |  |      |  |      |     |       uint32
      |  |  |      |  |      |     +--ro (type)?
      |  |  |      |  |      |        +--:(numbered-node-hop)
      |  |  |      |  |      |        |  +--ro numbered-node-hop
```

```
      |  |  |  |  |  |  |  |     +--ro node-id    te-node-id
      |  |  |  |  |  |  |  |     +--ro flags*
      |  |  |  |  |  |  |  |             path-attribute-flags
      |  |  |  |  |  |  |  +--:(numbered-link-hop)
      |  |  |  |  |  |  |  |  +--ro numbered-link-hop
      |  |  |  |  |  |  |  |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |  |  |     +--ro flags*
      |  |  |  |  |  |  |  |             path-attribute-flags
      |  |  |  |  |  |  |  +--:(unnumbered-link-hop)
      |  |  |  |  |  |  |  |  +--ro unnumbered-link-hop
      |  |  |  |  |  |  |  |     +--ro link-tp-id    te-tp-id
      |  |  |  |  |  |  |  |     +--ro node-id?
      |  |  |  |  |  |  |  |     |       te-node-id
      |  |  |  |  |  |  |  |     +--ro flags*
      |  |  |  |  |  |  |  |             path-attribute-flags
      |  |  |  |  |  |  |  +--:(label)
      |  |  |  |  |  |  |     +--ro label-hop
      |  |  |  |  |  |  |        +--ro te-label
      |  |  |  |  |  |  |        |  +--ro (technology)?
      |  |  |  |  |  |  |        |  |  +--:(generic)
      |  |  |  |  |  |  |        |  |     +--ro generic?
      |  |  |  |  |  |  |        |  rt-types:generalized-label
      |  |  |  |  |  |  |        |  +--ro direction?
      |  |  |  |  |  |  |        |          te-label-direction
      |  |  |  |  |  |  |        +--ro flags*
      |  |  |  |  |  |  |                path-attribute-flags
      |  |  |  |  |  +--ro path-properties
      |  |  |  |  |     +--ro path-metric* [metric-type]
      |  |  |  |  |     |  +--ro metric-type
      |  |  |  |  |     |  |       identityref
      |  |  |  |  |     |  +--ro accumulative-value?   uint64
      |  |  |  |  |     +--ro path-affinities-values
      |  |  |  |  |     |  +--ro path-affinities-value* [usage]
      |  |  |  |  |     |     +--ro usage    identityref
      |  |  |  |  |     |     +--ro value?   admin-groups
      |  |  |  |  |     +--ro path-affinity-names
      |  |  |  |  |     |  +--ro path-affinity-name* [usage]
      |  |  |  |  |     |     +--ro usage            identityref
      |  |  |  |  |     |     +--ro affinity-name* [name]
      |  |  |  |  |     |        +--ro name    string
      |  |  |  |  |     +--ro path-srlgs-lists
      |  |  |  |  |     |  +--ro path-srlgs-list* [usage]
      |  |  |  |  |     |     +--ro usage     identityref
      |  |  |  |  |     |     +--ro values*   srlg
      |  |  |  |  |     +--ro path-srlgs-names
      |  |  |  |  |     |  +--ro path-srlgs-name* [usage]
      |  |  |  |  |     |     +--ro usage    identityref
      |  |  |  |  |     |     +--ro names*   string
```

```
      |   |   |       |   |          +--ro path-route-objects
      |   |   |       |   |          |  +--ro path-computed-route-object*
      |   |   |       |   |          |           [index]
      |   |   |       |   |          |     +--ro index
      |   |   |       |   |          |     |       uint32
      |   |   |       |   |          |     +--ro (type)?
      |   |   |       |   |          |        +--:(numbered-node-hop)
      |   |   |       |   |          |        |  +--ro numbered-node-hop
      |   |   |       |   |          |        |     +--ro node-id
      |   |   |       |   |          |        |     |       te-node-id
      |   |   |       |   |          |        |     +--ro hop-type?
      |   |   |       |   |          |        |               te-hop-type
      |   |   |       |   |          |        +--:(numbered-link-hop)
      |   |   |       |   |          |        |  +--ro numbered-link-hop
      |   |   |       |   |          |        |     +--ro link-tp-id
      |   |   |       |   |          |        |     |       te-tp-id
      |   |   |       |   |          |        |     +--ro hop-type?
      |   |   |       |   |          |        |     |       te-hop-type
      |   |   |       |   |          |        |     +--ro direction?
      |   |   |       |   |          |        |               te-link-direction
      |   |   |       |   |          |        +--:(unnumbered-link-hop)
      |   |   |       |   |          |        |  +--ro unnumbered-link-hop
      |   |   |       |   |          |        |     +--ro link-tp-id
      |   |   |       |   |          |        |     |       te-tp-id
      |   |   |       |   |          |        |     +--ro node-id
      |   |   |       |   |          |        |     |       te-node-id
      |   |   |       |   |          |        |     +--ro hop-type?
      |   |   |       |   |          |        |     |       te-hop-type
      |   |   |       |   |          |        |     +--ro direction?
      |   |   |       |   |          |        |               te-link-direction
      |   |   |       |   |          |        +--:(as-number)
      |   |   |       |   |          |        |  +--ro as-number-hop
      |   |   |       |   |          |        |     +--ro as-number
      |   |   |       |   |          |        |     |       inet:as-number
      |   |   |       |   |          |        |     +--ro hop-type?
      |   |   |       |   |          |        |               te-hop-type
      |   |   |       |   |          |        +--:(label)
      |   |   |       |   |          |           +--ro label-hop
      |   |   |       |   |          |              +--ro te-label
      |   |   |       |   |          |                 +--ro (technology)?
      |   |   |       |   |          |                 |  +--:(generic)
      |   |   |       |   |          |                 |     +--ro generic?
      |   |   |       |   |          |                  rt-types:generalized-label
      |   |   |       |   |          |                    +--ro direction?
      |   |   |       |   |          |                   te-label-direction
      |   |   |       |   |          +--ro shared-resources-tunnels
      |   |   |       |   |             +--ro lsp-shared-resources-tunnel*
      |   |   |       |   |                     tunnel-ref
```

```
|  |  |  |  |  +--rw p2p-secondary-reverse-path
|  |  |  |  |     +--rw secondary-path?        leafref
|  |  |  |  |     +--rw path-setup-protocol?   identityref
|  |  |  |  +--rw candidate-p2p-secondary-paths
|  |  |  |     +--rw candidate-p2p-secondary-path*
|  |  |  |              [secondary-path]
|  |  |  |        +--rw secondary-path         leafref
|  |  |  |        +--rw path-setup-protocol?   identityref
|  |  |  |        +--ro active?                boolean
|  |  |  +--rw p2p-secondary-paths
|  |  |  |  +--rw p2p-secondary-path* [name]
|  |  |  |     +--rw name                                string
|  |  |  |     +--rw path-setup-protocol?                identityref
|  |  |  |     +--rw path-computation-method?            identityref
|  |  |  |     +--rw path-computation-server?
|  |  |  |     |       inet:ip-address
|  |  |  |     +--rw compute-only?                       empty
|  |  |  |     +--rw use-path-computation?               boolean
|  |  |  |     +--rw lockdown?                           empty
|  |  |  |     +--ro path-scope?                         identityref
|  |  |  |     +--rw optimizations
|  |  |  |     |  +--rw (algorithm)?
|  |  |  |     |     +--:(metric) {path-optimization-metric}?
|  |  |  |     |        +--rw optimization-metric* [metric-type]
|  |  |  |     |        |  +--rw metric-type
|  |  |  |     |        |  |       identityref
|  |  |  |     |        |  +--rw weight?
|  |  |  |     |        |  |       uint8
|  |  |  |     |        |  +--rw explicit-route-exclude-objects
|  |  |  |     |        |  |  +--rw route-object-exclude-object*
|  |  |  |     |        |  |           [index]
|  |  |  |     |        |  |     +--rw index
|  |  |  |     |        |  |     |       uint32
|  |  |  |     |        |  |     +--rw (type)?
|  |  |  |     |        |  |        +--:(numbered-node-hop)
|  |  |  |     |        |  |        |  +--rw numbered-node-hop
|  |  |  |     |        |  |        |     +--rw node-id
|  |  |  |     |        |  |        |     |       te-node-id
|  |  |  |     |        |  |        |     +--rw hop-type?
|  |  |  |     |        |  |        |             te-hop-type
|  |  |  |     |        |  |        +--:(numbered-link-hop)
|  |  |  |     |        |  |        |  +--rw numbered-link-hop
|  |  |  |     |        |  |        |     +--rw link-tp-id
|  |  |  |     |        |  |        |     |       te-tp-id
|  |  |  |     |        |  |        |     +--rw hop-type?
|  |  |  |     |        |  |        |     |       te-hop-type
|  |  |  |     |        |  |        |     +--rw direction?
|  |  |  |     |        |  |        |             te-link-direction
```


+--:(unnumbered-link-hop)
+--rw unnumbered-link-hop
+--rw link-tp-id
te-tp-id
+--rw node-id
te-node-id
+--rw hop-type?
te-hop-type
+--rw direction?
te-link-direction
+--:(as-number)
+--rw as-number-hop
+--rw as-number
inet:as-number
+--rw hop-type?
te-hop-type
+--:(label)
+--rw label-hop
+--rw te-label
+--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
      |  |  |      +--rw preference?                          uint8
      |  |  |      +--rw k-requested-paths?                   uint8
      |  |  |      +--rw named-path-constraint?               leafref
      |  |  |      |       {te-types:named-path-constraints}?
      |  |  |      +--rw te-bandwidth
      |  |  |      |  +--rw (technology)?
      |  |  |      |     +--:(generic)
      |  |  |      |        +--rw generic?   te-bandwidth
      |  |  |      +--rw link-protection?                     identityref
      |  |  |      +--rw setup-priority?                      uint8
      |  |  |      +--rw hold-priority?                       uint8
      |  |  |      +--rw signaling-type?                      identityref
      |  |  |      +--rw path-metric-bounds
      |  |  |      |  +--rw path-metric-bound* [metric-type]
      |  |  |      |     +--rw metric-type    identityref
```

```
      |   |   |      |     +--rw upper-bound?   uint64
      |   |   |      +--rw path-affinities-values
      |   |   |      |  +--rw path-affinities-value* [usage]
      |   |   |      |     +--rw usage    identityref
      |   |   |      |     +--rw value?   admin-groups
      |   |   |      +--rw path-affinity-names
      |   |   |      |  +--rw path-affinity-name* [usage]
      |   |   |      |     +--rw usage            identityref
      |   |   |      |     +--rw affinity-name* [name]
      |   |   |      |        +--rw name    string
      |   |   |      +--rw path-srlgs-lists
      |   |   |      |  +--rw path-srlgs-list* [usage]
      |   |   |      |     +--rw usage     identityref
      |   |   |      |     +--rw values*   srlg
      |   |   |      +--rw path-srlgs-names
      |   |   |      |  +--rw path-srlgs-name* [usage]
      |   |   |      |     +--rw usage    identityref
      |   |   |      |     +--rw names*   string
      |   |   |      +--rw disjointness?
      |   |   |      |       te-path-disjointness
      |   |   |      +--rw explicit-route-objects-always
      |   |   |      |  +--rw route-object-exclude-always* [index]
      |   |   |      |  |  +--rw index                          uint32
      |   |   |      |  |  +--rw (type)?
      |   |   |      |  |     +--:(numbered-node-hop)
      |   |   |      |  |     |  +--rw numbered-node-hop
      |   |   |      |  |     |     +--rw node-id     te-node-id
      |   |   |      |  |     |     +--rw hop-type?   te-hop-type
      |   |   |      |  |     +--:(numbered-link-hop)
      |   |   |      |  |     |  +--rw numbered-link-hop
      |   |   |      |  |     |     +--rw link-tp-id    te-tp-id
      |   |   |      |  |     |     +--rw hop-type?     te-hop-type
      |   |   |      |  |     |     +--rw direction?    te-link-direction
      |   |   |      |  |     +--:(unnumbered-link-hop)
      |   |   |      |  |     |  +--rw unnumbered-link-hop
      |   |   |      |  |     |     +--rw link-tp-id    te-tp-id
      |   |   |      |  |     |     +--rw node-id       te-node-id
      |   |   |      |  |     |     +--rw hop-type?     te-hop-type
      |   |   |      |  |     |     +--rw direction?    te-link-direction
      |   |   |      |  |     +--:(as-number)
      |   |   |      |  |     |  +--rw as-number-hop
      |   |   |      |  |     |     +--rw as-number    inet:as-number
      |   |   |      |  |     |     +--rw hop-type?    te-hop-type
      |   |   |      |  |     +--:(label)
      |   |   |      |  |        +--rw label-hop
      |   |   |      |  |           +--rw te-label
      |   |   |      |  |              +--rw (technology)?
      |   |   |      |  |              |  +--:(generic)
```

```
      |   |   |      |   |               |     +--rw generic?
      |   |   |      |   |               |       rt-types:generalized-label
      |   |   |      |   |               +--rw direction?
      |   |   |      |   |                     te-label-direction
      |   |   |      |   +--rw route-object-include-exclude* [index]
      |   |   |      |      +--rw explicit-route-usage?        identityref
      |   |   |      |      +--rw index                        uint32
      |   |   |      |      +--rw (type)?
      |   |   |      |         +--:(numbered-node-hop)
      |   |   |      |         |  +--rw numbered-node-hop
      |   |   |      |         |     +--rw node-id     te-node-id
      |   |   |      |         |     +--rw hop-type?   te-hop-type
      |   |   |      |         +--:(numbered-link-hop)
      |   |   |      |         |  +--rw numbered-link-hop
      |   |   |      |         |     +--rw link-tp-id    te-tp-id
      |   |   |      |         |     +--rw hop-type?     te-hop-type
      |   |   |      |         |     +--rw direction?    te-link-direction
      |   |   |      |         +--:(unnumbered-link-hop)
      |   |   |      |         |  +--rw unnumbered-link-hop
      |   |   |      |         |     +--rw link-tp-id    te-tp-id
      |   |   |      |         |     +--rw node-id       te-node-id
      |   |   |      |         |     +--rw hop-type?     te-hop-type
      |   |   |      |         |     +--rw direction?    te-link-direction
      |   |   |      |         +--:(as-number)
      |   |   |      |         |  +--rw as-number-hop
      |   |   |      |         |     +--rw as-number    inet:as-number
      |   |   |      |         |     +--rw hop-type?    te-hop-type
      |   |   |      |         +--:(label)
      |   |   |      |         |  +--rw label-hop
      |   |   |      |         |     +--rw te-label
      |   |   |      |         |        +--rw (technology)?
      |   |   |      |         |        |  +--:(generic)
      |   |   |      |         |        |     +--rw generic?
      |   |   |      |         |        |     rt-types:generalized-label
      |   |   |      |         |        +--rw direction?
      |   |   |      |         |                 te-label-direction
      |   |   |      |         +--:(srlg)
      |   |   |      |            +--rw srlg
      |   |   |      |               +--rw srlg?   uint32
      |   |   |      +--rw shared-resources-tunnels
      |   |   |      |  +--rw lsp-shared-resources-tunnel*   tunnel-ref
      |   |   |      +--rw path-in-segment!
      |   |   |      |  +--rw label-restrictions
      |   |   |      |     +--rw label-restriction* [index]
      |   |   |      |        +--rw restriction?    enumeration
      |   |   |      |        +--rw index           uint32
      |   |   |      |        +--rw label-start
      |   |   |      |        |  +--rw te-label
```

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

```
      |   |   |       |  +--rw protection-reversion-disable?   boolean
      |   |   |       |  +--rw hold-off-time?                  uint32
      |   |   |       |  +--rw wait-to-revert?                 uint16
      |   |   |       |  +--rw aps-signal-id?                  uint8
      |   |   |       +--rw restoration
      |   |   |       |  +--rw enable?                          boolean
      |   |   |       |  +--rw restoration-type?                identityref
      |   |   |       |  +--rw restoration-scheme?              identityref
      |   |   |       |  +--rw restoration-reversion-disable?   boolean
      |   |   |       |  +--rw hold-off-time?                   uint32
      |   |   |       |  +--rw wait-to-restore?                 uint16
      |   |   |       |  +--rw wait-to-revert?                  uint16
      |   |   |       +--ro computed-paths-properties
      |   |   |       |  +--ro computed-path-properties* [k-index]
      |   |   |       |     +--ro k-index            uint8
      |   |   |       |     +--ro path-properties
      |   |   |       |        +--ro path-metric* [metric-type]
      |   |   |       |        |  +--ro metric-type           identityref
      |   |   |       |        |  +--ro accumulative-value?   uint64
      |   |   |       |        +--ro path-affinities-values
      |   |   |       |        |  +--ro path-affinities-value* [usage]
      |   |   |       |        |     +--ro usage    identityref
      |   |   |       |        |     +--ro value?   admin-groups
      |   |   |       |        +--ro path-affinity-names
      |   |   |       |        |  +--ro path-affinity-name* [usage]
      |   |   |       |        |     +--ro usage            identityref
      |   |   |       |        |     +--ro affinity-name* [name]
      |   |   |       |        |        +--ro name    string
      |   |   |       |        +--ro path-srlgs-lists
      |   |   |       |        |  +--ro path-srlgs-list* [usage]
      |   |   |       |        |     +--ro usage     identityref
      |   |   |       |        |     +--ro values*   srlg
      |   |   |       |        +--ro path-srlgs-names
      |   |   |       |        |  +--ro path-srlgs-name* [usage]
      |   |   |       |        |     +--ro usage    identityref
      |   |   |       |        |     +--ro names*   string
      |   |   |       |        +--ro path-route-objects
      |   |   |       |        |  +--ro path-computed-route-object* [index]
      |   |   |       |        |     +--ro index
      |   |   |       |        |     |       uint32
      |   |   |       |        |     +--ro (type)?
      |   |   |       |        |        +--:(numbered-node-hop)
      |   |   |       |        |        |  +--ro numbered-node-hop
      |   |   |       |        |        |     +--ro node-id     te-node-id
      |   |   |       |        |        |     +--ro hop-type?   te-hop-type
      |   |   |       |        |        +--:(numbered-link-hop)
      |   |   |       |        |        |  +--ro numbered-link-hop
      |   |   |       |        |        |     +--ro link-tp-id    te-tp-id
```

```
      |  |  |      |         |         |     +--ro hop-type?
      |  |  |      |         |         |     |       te-hop-type
      |  |  |      |         |         |     +--ro direction?
      |  |  |      |         |         |             te-link-direction
      |  |  |      |         |         +--:(unnumbered-link-hop)
      |  |  |      |         |         |  +--ro unnumbered-link-hop
      |  |  |      |         |         |     +--ro link-tp-id    te-tp-id
      |  |  |      |         |         |     +--ro node-id
      |  |  |      |         |         |     |       te-node-id
      |  |  |      |         |         |     +--ro hop-type?
      |  |  |      |         |         |     |       te-hop-type
      |  |  |      |         |         |     +--ro direction?
      |  |  |      |         |         |             te-link-direction
      |  |  |      |         |         +--:(as-number)
      |  |  |      |         |         |  +--ro as-number-hop
      |  |  |      |         |         |     +--ro as-number
      |  |  |      |         |         |     |       inet:as-number
      |  |  |      |         |         |     +--ro hop-type?
      |  |  |      |         |         |             te-hop-type
      |  |  |      |         |         +--:(label)
      |  |  |      |         |            +--ro label-hop
      |  |  |      |         |               +--ro te-label
      |  |  |      |         |                  +--ro (technology)?
      |  |  |      |         |                  |  +--:(generic)
      |  |  |      |         |                  |     +--ro generic?
      |  |  |      |         |                  | rt-types:generalized-label
      |  |  |      |         |                  +--ro direction?
      |  |  |      |         |                          te-label-direction
      |  |  |      |         +--ro shared-resources-tunnels
      |  |  |      |            +--ro lsp-shared-resources-tunnel*
      |  |  |      |                    tunnel-ref
      |  |  |      +--ro computed-path-error-infos
      |  |  |      |  +--ro computed-path-error-info* []
      |  |  |      |     +--ro error-description?   string
      |  |  |      |     +--ro error-timestamp?     yang:date-and-time
      |  |  |      |     +--ro error-reason?        identityref
      |  |  |      +--ro lsp-provisioning-error-infos
      |  |  |      |  +--ro lsp-provisioning-error-info* []
      |  |  |      |     +--ro error-description?   string
      |  |  |      |     +--ro error-timestamp?     yang:date-and-time
      |  |  |      |     +--ro error-node-id?       te-types:te-node-id
      |  |  |      |     +--ro error-link-id?       te-types:te-tp-id
      |  |  |      |     +--ro lsp-id?              uint16
      |  |  |      +--ro lsps
      |  |  |         +--ro lsp* [lsp-id]
      |  |  |            +--ro lsp-provisioning-error-infos
      |  |  |            |  +--ro lsp-provisioning-error-info* []
      |  |  |            |     +--ro error-description?   string
```

```
|   |   |           |     +--ro error-timestamp?
|   |   |           |     |       yang:date-and-time
|   |   |           |     +--ro error-node-id?
|   |   |           |     |       te-types:te-node-id
|   |   |           |     +--ro error-link-id?
|   |   |           |             te-types:te-tp-id
|   |   |           +--ro source?
|   |   |           |       te-types:te-node-id
|   |   |           +--ro destination?
|   |   |           |       te-types:te-node-id
|   |   |           +--ro tunnel-id?
|   |   |           |       uint16
|   |   |           +--ro lsp-id
|   |   |           |       uint16
|   |   |           +--ro extended-tunnel-id?
|   |   |           |       yang:dotted-quad
|   |   |           +--ro operational-state?
|   |   |           |       identityref
|   |   |           +--ro path-setup-protocol?
|   |   |           |       identityref
|   |   |           +--ro origin-type?
|   |   |           |       enumeration
|   |   |           +--ro lsp-resource-status?
|   |   |           |       enumeration
|   |   |           +--ro lockout-of-normal?
|   |   |           |       boolean
|   |   |           +--ro freeze?
|   |   |           |       boolean
|   |   |           +--ro lsp-protection-role?
|   |   |           |       enumeration
|   |   |           +--ro lsp-protection-state?
|   |   |           |       identityref
|   |   |           +--ro protection-group-ingress-node-id?
|   |   |           |       te-types:te-node-id
|   |   |           +--ro protection-group-egress-node-id?
|   |   |           |       te-types:te-node-id
|   |   |           +--ro lsp-shared-resources-tunnel?
|   |   |           |       tunnel-ref
|   |   |           +--ro lsp-record-route-information
|   |   |           |  +--ro lsp-record-route-information* [index]
|   |   |           |     +--ro index                          uint32
|   |   |           |     +--ro (type)?
|   |   |           |        +--:(numbered-node-hop)
|   |   |           |        |  +--ro numbered-node-hop
|   |   |           |        |     +--ro node-id    te-node-id
|   |   |           |        |     +--ro flags*
|   |   |           |        |              path-attribute-flags
|   |   |           |        +--:(numbered-link-hop)
```

```
      |  |  |           |         |  +--ro numbered-link-hop
      |  |  |           |         |     +--ro link-tp-id    te-tp-id
      |  |  |           |         |     +--ro flags*
      |  |  |           |         |              path-attribute-flags
      |  |  |           |         +--:(unnumbered-link-hop)
      |  |  |           |         |  +--ro unnumbered-link-hop
      |  |  |           |         |     +--ro link-tp-id    te-tp-id
      |  |  |           |         |     +--ro node-id?      te-node-id
      |  |  |           |         |     +--ro flags*
      |  |  |           |         |              path-attribute-flags
      |  |  |           |         +--:(label)
      |  |  |           |            +--ro label-hop
      |  |  |           |               +--ro te-label
      |  |  |           |               |  +--ro (technology)?
      |  |  |           |               |  |  +--:(generic)
      |  |  |           |               |  |     +--ro generic?
      |  |  |           |               |  |  rt-types:generalized-label
      |  |  |           |               |  +--ro direction?
      |  |  |           |               |           te-label-direction
      |  |  |           |               +--ro flags*
      |  |  |           |                        path-attribute-flags
      |  |  |           +--ro path-properties
      |  |  |           |  +--ro path-metric* [metric-type]
      |  |  |           |  |  +--ro metric-type           identityref
      |  |  |           |  |  +--ro accumulative-value?   uint64
      |  |  |           |  +--ro path-affinities-values
      |  |  |           |  |  +--ro path-affinities-value* [usage]
      |  |  |           |  |     +--ro usage    identityref
      |  |  |           |  |     +--ro value?   admin-groups
      |  |  |           |  +--ro path-affinity-names
      |  |  |           |  |  +--ro path-affinity-name* [usage]
      |  |  |           |  |     +--ro usage            identityref
      |  |  |           |  |     +--ro affinity-name* [name]
      |  |  |           |  |        +--ro name    string
      |  |  |           |  +--ro path-srlgs-lists
      |  |  |           |  |  +--ro path-srlgs-list* [usage]
      |  |  |           |  |     +--ro usage     identityref
      |  |  |           |  |     +--ro values*   srlg
      |  |  |           |  +--ro path-srlgs-names
      |  |  |           |  |  +--ro path-srlgs-name* [usage]
      |  |  |           |  |     +--ro usage    identityref
      |  |  |           |  |     +--ro names*   string
      |  |  |           |  +--ro path-route-objects
      |  |  |           |  |  +--ro path-computed-route-object* [index]
      |  |  |           |  |     +--ro index
      |  |  |           |  |     |       uint32
      |  |  |           |  |     +--ro (type)?
      |  |  |           |  |        +--:(numbered-node-hop)
```

```
      |  |  |           |  |         |  +--ro numbered-node-hop
      |  |  |           |  |         |     +--ro node-id     te-node-id
      |  |  |           |  |         |     +--ro hop-type?   te-hop-type
      |  |  |           |  |         +--:(numbered-link-hop)
      |  |  |           |  |         |  +--ro numbered-link-hop
      |  |  |           |  |         |     +--ro link-tp-id    te-tp-id
      |  |  |           |  |         |     +--ro hop-type?
      |  |  |           |  |         |     |       te-hop-type
      |  |  |           |  |         |     +--ro direction?
      |  |  |           |  |         |             te-link-direction
      |  |  |           |  |         +--:(unnumbered-link-hop)
      |  |  |           |  |         |  +--ro unnumbered-link-hop
      |  |  |           |  |         |     +--ro link-tp-id    te-tp-id
      |  |  |           |  |         |     +--ro node-id
      |  |  |           |  |         |     |       te-node-id
      |  |  |           |  |         |     +--ro hop-type?
      |  |  |           |  |         |     |       te-hop-type
      |  |  |           |  |         |     +--ro direction?
      |  |  |           |  |         |             te-link-direction
      |  |  |           |  |         +--:(as-number)
      |  |  |           |  |         |  +--ro as-number-hop
      |  |  |           |  |         |     +--ro as-number
      |  |  |           |  |         |     |       inet:as-number
      |  |  |           |  |         |     +--ro hop-type?
      |  |  |           |  |         |             te-hop-type
      |  |  |           |  |         +--:(label)
      |  |  |           |  |            +--ro label-hop
      |  |  |           |  |               +--ro te-label
      |  |  |           |  |                  +--ro (technology)?
      |  |  |           |  |                  |  +--:(generic)
      |  |  |           |  |                  |     +--ro generic?
      |  |  |           |  |                  | rt-types:generalized-label
      |  |  |           |  |                  +--ro direction?
      |  |  |           |  |                          te-label-direction
      |  |  |           |  +--ro shared-resources-tunnels
      |  |  |           |     +--ro lsp-shared-resources-tunnel*
      |  |  |           |             tunnel-ref
      |  |  |           +--ro te-dev:lsp-timers
      |  |  |           |  +--ro te-dev:life-time?         uint32
      |  |  |           |  +--ro te-dev:time-to-install?   uint32
      |  |  |           |  +--ro te-dev:time-to-destroy?   uint32
      |  |  |           +--ro te-dev:downstream-info
      |  |  |           |  +--ro te-dev:nhop?
      |  |  |           |  |       inet:ip-address
      |  |  |           |  +--ro te-dev:outgoing-interface?
      |  |  |           |  |       if:interface-ref
      |  |  |           |  +--ro te-dev:neighbor?
      |  |  |           |  |       inet:ip-address
```

```
|   |   |           |  +--ro te-dev:label?
|   |   |           |          rt-types:generalized-label
|   |   |           +--ro te-dev:upstream-info
|   |   |              +--ro te-dev:phop?       inet:ip-address
|   |   |              +--ro te-dev:neighbor?   inet:ip-address
|   |   |              +--ro te-dev:label?
|   |   |                      rt-types:generalized-label
|   |   +---x tunnel-action
|   |   |  +---w input
|   |   |  |  +---w action-type?   identityref
|   |   |  +--ro output
|   |   |     +--ro action-result?   identityref
|   |   +---x protection-external-commands
|   |   |  +---w input
|   |   |     +---w protection-external-command?        identityref
|   |   |     +---w protection-group-ingress-node-id?
|   |   |     |       te-types:te-node-id
|   |   |     +---w protection-group-egress-node-id?
|   |   |     |       te-types:te-node-id
|   |   |     +---w path-ref?                           path-ref
|   |   |     +---w traffic-type?                       enumeration
|   |   |     +---w extra-traffic-tunnel-ref?           tunnel-ref
|   |   +--rw te-dev:lsp-install-interval?        uint32
|   |   +--rw te-dev:lsp-cleanup-interval?        uint32
|   |   +--rw te-dev:lsp-invalidation-interval?   uint32
|   +--rw tunnel-p2mp* [name]
|      +--rw name                  string
|      +--rw identifier?           uint16
|      +--rw description?          string
|      +--ro operational-state?    identityref
+--ro lsps-state
|  +--ro lsp*
|           [source destination tunnel-id lsp-id extended-tunnel-id]
|      +--ro source
|      |       te-types:te-node-id
|      +--ro destination
|      |       te-types:te-node-id
|      +--ro tunnel-id                          uint16
|      +--ro lsp-id                             uint16
|      +--ro extended-tunnel-id                 yang:dotted-quad
|      +--ro operational-state?                 identityref
|      +--ro path-setup-protocol?               identityref
|      +--ro origin-type?                       enumeration
|      +--ro lsp-resource-status?               enumeration
|      +--ro lockout-of-normal?                 boolean
|      +--ro freeze?                            boolean
|      +--ro lsp-protection-role?               enumeration
|      +--ro lsp-protection-state?              identityref
```

```
|      +--ro protection-group-ingress-node-id?
|      |       te-types:te-node-id
|      +--ro protection-group-egress-node-id?
|      |       te-types:te-node-id
|      +--ro lsp-record-route-information
|      |  +--ro lsp-record-route-information* [index]
|      |     +--ro index                          uint32
|      |     +--ro (type)?
|      |        +--:(numbered-node-hop)
|      |        |  +--ro numbered-node-hop
|      |        |     +--ro node-id    te-node-id
|      |        |     +--ro flags*     path-attribute-flags
|      |        +--:(numbered-link-hop)
|      |        |  +--ro numbered-link-hop
|      |        |     +--ro link-tp-id    te-tp-id
|      |        |     +--ro flags*        path-attribute-flags
|      |        +--:(unnumbered-link-hop)
|      |        |  +--ro unnumbered-link-hop
|      |        |     +--ro link-tp-id    te-tp-id
|      |        |     +--ro node-id?      te-node-id
|      |        |     +--ro flags*        path-attribute-flags
|      |        +--:(label)
|      |           +--ro label-hop
|      |              +--ro te-label
|      |              |  +--ro (technology)?
|      |              |  |  +--:(generic)
|      |              |  |     +--ro generic?
|      |              |  |              rt-types:generalized-label
|      |              |  +--ro direction?       te-label-direction
|      |              +--ro flags*      path-attribute-flags
|      +--ro te-dev:lsp-timers
|      |  +--ro te-dev:life-time?         uint32
|      |  +--ro te-dev:time-to-install?   uint32
|      |  +--ro te-dev:time-to-destroy?   uint32
|      +--ro te-dev:downstream-info
|      |  +--ro te-dev:nhop?                  inet:ip-address
|      |  +--ro te-dev:outgoing-interface?   if:interface-ref
|      |  +--ro te-dev:neighbor?              inet:ip-address
|      |  +--ro te-dev:label?
|      |           rt-types:generalized-label
|      +--ro te-dev:upstream-info
|         +--ro te-dev:phop?       inet:ip-address
|         +--ro te-dev:neighbor?   inet:ip-address
|         +--ro te-dev:label?      rt-types:generalized-label
|    +--rw te-dev:interfaces
|    |  +--rw te-dev:threshold-type?            enumeration
|    |  +--rw te-dev:delta-percentage?          rt-types:percentage
|    |  +--rw te-dev:threshold-specification?   enumeration
```

```
|  +--rw te-dev:up-thresholds*             rt-types:percentage
|  +--rw te-dev:down-thresholds*           rt-types:percentage
|  +--rw te-dev:up-down-thresholds*        rt-types:percentage
|  +--rw te-dev:interface* [interface]
|     +--rw te-dev:interface
|     |       if:interface-ref
|     +--rw te-dev:te-metric?
|     |       te-types:te-metric
|     +--rw (te-dev:admin-group-type)?
|     |  +--:(te-dev:value-admin-groups)
|     |  |  +--rw (te-dev:value-admin-group-type)?
|     |  |     +--:(te-dev:admin-groups)
|     |  |     |  +--rw te-dev:admin-group?
|     |  |     |           te-types:admin-group
|     |  |     +--:(te-dev:extended-admin-groups)
|     |  |              {te-types:extended-admin-groups}?
|     |  |        +--rw te-dev:extended-admin-group?
|     |  |                 te-types:extended-admin-group
|     |  +--:(te-dev:named-admin-groups)
|     |     +--rw te-dev:named-admin-groups* [named-admin-group]
|     |        +--rw te-dev:named-admin-group    leafref
|     +--rw (te-dev:srlg-type)?
|     |  +--:(te-dev:value-srlgs)
|     |  |  +--rw te-dev:values* [value]
|     |  |     +--rw te-dev:value    uint32
|     |  +--:(te-dev:named-srlgs)
|     |     +--rw te-dev:named-srlgs* [named-srlg]
|     |              {te-types:named-srlg-groups}?
|     |        +--rw te-dev:named-srlg    leafref
|     +--rw te-dev:threshold-type?
|     |       enumeration
|     +--rw te-dev:delta-percentage?
|     |       rt-types:percentage
|     +--rw te-dev:threshold-specification?
|     |       enumeration
|     +--rw te-dev:up-thresholds*
|     |       rt-types:percentage
|     +--rw te-dev:down-thresholds*
|     |       rt-types:percentage
|     +--rw te-dev:up-down-thresholds*
|     |       rt-types:percentage
|     +--rw te-dev:switching-capabilities* [switching-capability]
|     |  +--rw te-dev:switching-capability    identityref
|     |  +--rw te-dev:encoding?               identityref
|     +--ro te-dev:state
|        +--ro te-dev:te-advertisements-state
|           +--ro te-dev:flood-interval?           uint32
|           +--ro te-dev:last-flooded-time?        uint32
|
```

```
      |           +--ro te-dev:next-flooded-time?        uint32
      |           +--ro te-dev:last-flooded-trigger?     enumeration
      |           +--ro te-dev:advertized-level-areas* [level-area]
      |              +--ro te-dev:level-area    uint32
     +--rw te-dev:performance-thresholds

  rpcs:
    +---x globals-rpc
    +---x interfaces-rpc
    +---x tunnels-rpc
       +---w input
       |  +---w tunnel-info
       |     +---w (type)?
       |        +--:(tunnel-p2p)
       |        |  +---w p2p-id?    tunnel-ref
       |        +--:(tunnel-p2mp)
       |           +---w p2mp-id?   tunnel-p2mp-ref
       +--ro output
          +--ro result
             +--ro result?   enumeration

  notifications:
    +---n globals-notif
    +---n tunnels-notif
module: ietf-te-device

  rpcs:
    +---x interfaces-rpc

  notifications:
    +---n interfaces-notif
```

Figure 3: TE generic model configuration and state tree

3. Model Organization

The TE generic YANG data module "ietf-te" covers configuration, state, RPC and notifications data pertaining to TE global, tunnels and LSPs parameters that are device independent.

The container "te" is the top level container in the data model. The presence of this container enables TE function system wide.

The model top level organization is shown below in Figure 4:

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

      +--rw tunnels
         .
         .

      +-- lsps-state

rpcs:
   +---x globals-rpc
   +---x tunnels-rpc
notifications:
   +---n globals-notif
   +---n tunnels-notif
```

Figure 4: TE generic highlevel model view

3.1. Global Configuration and State Data

The global TE branch of the data model covers configurations that control TE features behavior system-wide, and its respective state. Examples of such configuration data are:

- Table of named SRLG mappings
- Table of named (extended) administrative groups mappings
- Table of named path-constraints sets
- System-wide capabilities for LSP reoptimization
  - Reoptimization timers (periodic interval, LSP installation and cleanup)
  - Link state flooding thresholds
  - Periodic flooding interval
- Global capabilities that affect originating, transiting and terminating LSPs. For example:
  - Path selection parameters (e.g. metric to optimize, etc.)
  - Path or segment protection parameters

3.2. Interfaces Configuration and State Data

This branch of the model covers configuration and state data corresponding to TE interfaces that are present on a device. The module "ietf-te-device" is introduced to hold such TE device specific properties.

Examples of TE interface properties are: * Maximum reservable bandwidth, bandwidth constraints (BC) * Flooding parameters * Flooding intervals and threshold values * interface attributes * (Extended) administrative groups * SRLG values * TE metric value * Fast reroute backup tunnel properties (such as static, auto-tunnel)

```
module: ietf-te-device
  augment /te:te:
      +--rw interfaces
         .
         +-- rw te-dev:te-attributes
                <<intended configuration>>
             .
             +-- ro state
                <<derived state associated with the TE interface>>
```

Figure 5: TE interface state

The derived state associated with interfaces is grouped under the interface "state" sub-container as shown in Figure 5. This covers state data such as:

- Bandwidth information: maximum bandwidth, available bandwidth at different priorities and for each class-type (CT)
- List of admitted LSPs
  - Name, bandwidth value and pool, time, priority
- Statistics: state counters, flooding counters, admission counters (accepted/rejected), preemption counters
- Adjacency information
  - Neighbor address
  - Metric value

3.3. Tunnels Configuration and State Data

This branch covers data related to TE tunnels configuration and state. The derived state associated with tunnels is grouped under a state container as shown in Figure 6.

```
module: ietf-te
   +--rw te!
      +--rw tunnels
            <<intended configuration>>
         .
         +-- ro state
            <<derived state associated with the tunnel>>
```

Figure 6: TE interface state tree

Examples of tunnel configuration data for TE tunnels:

- Name and type (e.g. P2P, P2MP) of the TE tunnel
- Administrative and operational state of the TE tunnel
- Set of primary and corresponding secondary paths and corresponding path attributes
- Bidirectional path attribute(s) including forwarding and reverse path properties
- Protection and restoration path parameters

3.3.1. Tunnel Compute-Only Mode

A configured TE tunnel, by default, is provisioned so it can carry traffic as soon as a valid path is computed and an LSP instantiated. In some cases, however, a TE tunnel may be provisioned for the only purpose of computing a path and reporting it without the need to instantiate the LSP or commit any resources. In such a case, the tunnel is configured in "compute-only" mode to distinguish it from default tunnel behavior.

A "compute-only" TE tunnel is configured as a usual TE tunnel with associated per path constraint(s) and properties on a device or controller. The device or controller computes the feasible path(s) subject to configured constraints and reflects the computed path(s) in the LSP(s) Record-Route Object (RRO) list. At any time, a client may query "on-demand" the "compute-only" TE tunnel computed path(s) properties by querying the state of the tunnel. Alternatively, the

client can subscribe on the "compute-only" TE tunnel to be notified of computed path(s) and whenever it changes.

3.3.2. Tunnel Hierarchical Link Endpoint

TE LSPs can be set up in MPLS or Generalized MPLS (GMPLS) networks to be used to form links to carry traffic in in other (client) networks [RFC6107]. In this case, the model introduces the TE tunnel hierarchical link endpoint parameters to identify the specific link in the client layer that the underlying TE tunnel is associated with.

3.4. TE LSPs State Data

TE LSPs are derived state data that are present whenever the LSP(s) are instantiated - for example, when associated signaling completes. TE LSPs exists on routers as ingress (starting point of LSP), transit (mid-point of LSP), or egress (termination point of the LSP). In the model, the nodes holding TE LSP data exist in the read-only lsps-state list as show in Figure 3.

3.5. Global RPC Data

This branch of the model covers system-wide RPC execution data to trigger actions and optionally expect responses. Examples of such TE commands are to:

- Clear global TE statistics of various features

3.6. Interface RPC Data

This collection of data in the model defines TE interface RPC execution commands. Examples of these are to:

- Clear TE statistics for all or for individual TE interfaces

- Trigger immediate flooding for one or all TE interfaces

3.7. Tunnel RPC Data

This branch of the model covers TE tunnel RPC execution data to trigger actions and expect responses. The TE generic YANG data model defines target containers that an external module in [I-D.ietf-teas-yang-path-computation] augments with RPCs that allow the invocation of certain TE functions (e.g. path computations).

4. TE Generic and Helper YANG Modules

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

- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]

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

```
<CODE BEGINS> file "ietf-te@2019-04-09.yang"
module ietf-te {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-te";

  /* Replace with IANA when assigned */
  prefix "te";

  /* Import TE generic types */
  import ietf-te-types {
    prefix te-types;
    reference "draft-ietf-teas-yang-te-types: A YANG Data Model for
               Common Traffic Engineering Types";
  }

  import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
  }

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

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

     WG Chair: Lou Berger
               <mailto:lberger@labn.net>
```

```
     WG Chair: Vishnu Pavan Beeram
               <mailto:vbeeram@juniper.net>

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

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

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

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

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

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

  description
    "YANG data module for TE configuration,
    state, RPC and notifications.";

  revision "2019-04-09" {
    description "Latest update to TE generic YANG module.";
    reference
      "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels
       and Interfaces";
  }

  identity path-computation-error-reason {
    description
      "Base identity for path computation error reasons";
  }
  identity path-computation-error-no-topology {
    base path-computation-error-reason;
    description
      "Path computation error no topology error reason";
  }
  identity path-computation-error-no-server {
    base path-computation-error-reason;
    description
      "Path computation error no server error reason";
  }
  identity path-computation-error-path-not-found {
    base path-computation-error-reason;
```

```
    description
      "Path computation no path found error reason";
  }

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

  typedef tunnel-p2mp-ref {
    type leafref {
      path "/te:te/te:tunnels/te:tunnel-p2mp/te:name";
    }
    description
      "This type is used by data models that need to reference
       configured P2MP TE tunnel.";
    reference "RFC4875";
  }

  typedef path-ref {
    type union {
      type leafref {
        path "/te:te/te:tunnels/te:tunnel/" +
              "te:p2p-primary-paths/te:p2p-primary-path/te:name";
      }
      type leafref {
        path "/te:te/te:tunnels/te:tunnel/" +
             "te:p2p-secondary-paths/te:p2p-secondary-path/te:name";
      }
    }
    description
      "This type is used by data models that need to reference
       configured primary or secondary path of a TE tunnel.";
  }

  /**
   * TE tunnel generic groupings
   */
  grouping p2p-secondary-path-properties {
    description "tunnel path properties.";
    uses p2p-path-properties;
    uses path-constraints-common;
    uses protection-restoration-properties;
    uses p2p-path-properties-state;
```

```
  }

  grouping p2p-primary-path-properties {
    description
      "TE tunnel primary path properties grouping";
    uses p2p-path-properties;
    uses path-constraints-common;
    uses p2p-path-properties-state;
  }

  grouping path-properties {
    description "TE computed path properties grouping";
    container path-properties {
      description "The TE path computed properties";
      list path-metric {
        key metric-type;
        description "TE path metric type";
        leaf metric-type {
          type identityref {
            base te-types:path-metric-type;
          }
          description "TE path metric type";
        }
        leaf accumulative-value {
          type uint64;
          description "TE path metric accumulative value";
        }
      }
      uses te-types:generic-path-affinities;
      uses te-types:generic-path-srlgs;
      container path-route-objects {
        config 'false';
        description
          "Container for the list of computed route objects
           as returned by the computation engine";
        list path-computed-route-object {
          key index;
          ordered-by user;
          description
            "List of computed route objects returned by the
             computation engine";
          leaf index {
            type uint32;
            description
              "Route object entry index. The index is used to
               identify an entry in the list. The order of entries
               is defined by the user without relying on key values";
          }
```

```
          uses te-types:explicit-route-hop;
        }
      }
      uses shared-resources-tunnels;
    }
  }

  grouping p2p-path-properties-state {
    description "TE per path state parameters";
    container computed-paths-properties {
      config 'false';
      description "Computed path properties container";
      list computed-path-properties {
        key k-index;
        description "List of computed paths";
        leaf k-index {
          type uint8;
          description
            "The k-th path returned from the computation server.
             A lower k value path is more optimal than higher k
             value path(s)";
        }
        uses path-properties {
          description "The TE path computed properties";
        }
      }
    }
    uses computed-path-error-info;
    uses lsp-provisioning-error-info {
      augment "lsp-provisioning-error-infos/" +
              "lsp-provisioning-error-info" {
        description
          "Augmentation of LSP provisioning information under a
           specific path";
        leaf lsp-id {
          type uint16;
          description
            "The LSP-ID for which path compatutation was performed.";
        }
      }
    }
    container lsps {
      config 'false';
      description "TE LSPs container";
      list lsp {
        key "lsp-id";
        description "List of LSPs associated with the tunnel.";
```

```
        uses lsp-provisioning-error-info;
        uses lsp-properties-state;
        uses shared-resources-tunnels-state;
        uses lsp-record-route-information-state;
        uses path-properties {
          description "The TE path actual properties";
        }
      }
    }
  }

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

  grouping lsp-provisioning-error-info {
    description
      "Grouping for LSP provisioning error information";
    container lsp-provisioning-error-infos {
      config false;
```

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

  grouping p2p-path-properties-common {
    description
      "TE tunnel common path properties configuration grouping";
    leaf name {
      type string;
      description "TE path name";
    }
    leaf path-setup-protocol {
      type identityref {
        base te-types:path-signaling-type;
      }
      default te-types:path-setup-static;
      description
        "Signaling protocol used to set up this tunnel";
    }
    leaf path-computation-method {
```

```
      type identityref {
        base te-types:path-computation-method;
      }
      default te-types:path-locally-computed;
      description
        "The method used for computing the path, either
        locally computed, queried from a server or not
        computed at all (explicitly configured).";
    }
    leaf path-computation-server {
      when "../path-computation-method = "+
      "'te-types:path-externally-queried'" {
        description
          "The path-computation server when the path is
           externally queried";
      }
      type inet:ip-address;
      description
        "Address of the external path computation
         server";
    }
    leaf compute-only {
      type empty;
      description
        "When set, the path is computed and updated whenever
         the topology is updated. No resources are committed
         or reserved in the network.";
    }
    leaf use-path-computation {
      when "../path-computation-method =" +
      " 'te-types:path-locally-computed'";
      type boolean;
      default 'true';
      description "A CSPF dynamically computed path";
    }
    leaf lockdown {
      type empty;
      description
        "Indicates no reoptimization to be attempted for
         this path.";
    }
    leaf path-scope {
      type identityref {
        base te-types:path-scope-type;
      }
      default te-types:path-scope-end-to-end;
      config 'false';
      description "Path scope if segment or an end-to-end path";
```

```
    }
  }

  grouping p2p-reverse-path-properties {
    description
      "TE tunnel reverse path properties configuration
       grouping";
    uses p2p-path-properties-common;
    uses te-types:generic-path-optimization;
    leaf named-path-constraint {
      if-feature te-types:named-path-constraints;
      type leafref {
        path "../../../../../../globals/"
        + "named-path-constraints/named-path-constraint/"
        + "name";
      }
      description
        "Reference to a globally defined named path
        constraint set";
    }
  }

  grouping p2p-primary-reverse-path-properties {
    description "TE P2P tunnel primary reverse path properties.";
    reference "RFC7551";
    container p2p-primary-reverse-path {
      description "Tunnel reverse primary path properties";
      uses p2p-reverse-path-properties;
      uses path-constraints-common;
      uses p2p-path-properties-state;
      container p2p-secondary-reverse-path {
        description "Tunnel reverse secondary path properties";
        uses p2p-secondary-reverse-path-properties;
      }
    }
  }

  grouping p2p-path-properties {
    description
      "TE tunnel path properties configuration grouping";
    uses p2p-path-properties-common;
    uses te-types:generic-path-optimization;
    leaf preference {
      type uint8 {
        range "1..255";
      }
      default 1;
      description
```

```
        "Specifies a preference for this path. The lower the
        number higher the preference";
    }
    leaf k-requested-paths {
      type uint8;
      default 1;
      description
        "The number of k-shortest-paths  requested from the path
         computation server and returned sorted by its optimization
         objective";
    }
    leaf named-path-constraint {
      if-feature te-types:named-path-constraints;
      type leafref {
        path "../../../../../globals/"
        + "named-path-constraints/named-path-constraint/"
        + "name";
      }
      description
        "Reference to a globally defined named path
        constraint set";
    }
  }

  grouping hierarchical-link-properties {
    description
      "Hierarchical link grouping";
    reference "RFC4206";
    container hierarchical-link {
      description
        "Identifies a hierarchical link (in client layer)
         that this tunnel is associated with.";
      leaf local-te-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
          "Local TE node identifier";
      }
      leaf local-te-link-tp-id {
        type te-types:te-tp-id;
        default 0;
        description
          "Local TE link termination point identifier";
      }
      leaf remote-te-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
```

```
          "Remote TE node identifier";
      }
      uses te-types:te-topology-identifier;
    }
  }

  grouping protection-restoration-properties-state {
    description
      "Protection parameters grouping";
    leaf lockout-of-normal {
      type boolean;
      default 'false';
      description
        "
          When set to 'True', it represents a lockout of normal
          traffic external command. When set to 'False', it
          represents a clear lockout of normal traffic external
          command. The lockout of normal traffic command applies
          to this Tunnel.
        ";
      reference "RFC4427";
    }
    leaf freeze {
      type boolean;
      default 'false';
      description
        "
          When set to 'True', it represents a freeze external
          command. When set to 'False', it represents a clear
          freeze external command. The freeze command command
          applies to all the Tunnels which are sharing the
          protection resources with this Tunnel.
        ";
      reference "RFC4427";
    }
    leaf lsp-protection-role {
      type enumeration {
        enum working {
          description
            "A working LSP must be a primary LSP whilst a protecting
             LSP can be either a primary or a secondary LSP. Also,
             known as protected LSPs when working LSPs are associated
             with protecting LSPs.";
        }
        enum protecting {
          description
            "A secondary LSP is an LSP that has been provisioned
             in the control plane only; e.g. resource allocation
```

```
              has not been committed at the data plane";
          }
        }
        default working;
        description "LSP role type";
        reference "RFC4872, section 4.2.1";
      }

      leaf lsp-protection-state {
        type identityref {
          base te-types:lsp-protection-state;
        }
        default te-types:normal;
        description
          "The state of the APS state machine controlling which
           tunnels is using the resources of the protecting LSP.";
      }
      leaf protection-group-ingress-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
          "Indicates the te-node-id of the protection group
          ingress node when the APS state represents an extenal
          command (LoP, SF, MS) applied to it or a WTR timer
          running on it. If the external command is not applied to
          the ingress node or the WTR timer is not running on it,
          this attribute is not specified. A value 0.0.0.0 is used
          when the te-node-id of the protection group ingress node is
          unknown (e.g., because the ingress node is outside the scope
          of control of the server)";
      }
      leaf protection-group-egress-node-id {
        type te-types:te-node-id;
        default "0.0.0.0";
        description
          "Indicates the te-node-id of the protection group egress node
          when the APS state represents an extenal command (LoP, SF,
          MS) applied to it or a WTR timer running on it. If the
          external command is not applied to the ingress node or
          the WTR timer is not running on it, this attribute is not
          specified. A value 0.0.0.0 is used when the te-node-id of
          the protection group ingress node is unknown (e.g., because
          the ingress node is outside the scope of control of the
          server)";
      }
    }

    grouping protection-restoration-properties {
```

```
    description "Protection and restoration parameters";
    container protection {
      description "Protection parameters";
      leaf enable {
        type boolean;
        default 'false';
        description
          "A flag to specify if LSP protection is enabled";
        reference "RFC4427";
      }
      leaf protection-type {
        type identityref {
          base te-types:lsp-protection-type;
        }
        default te-types:lsp-protection-unprotected;
        description "LSP protection type.";
      }
      leaf protection-reversion-disable {
        type boolean;
        default 'false';
        description "Disable protection reversion to working path";
      }
      leaf hold-off-time {
        type uint32;
        units "milli-seconds";
        default 0;
        description
          "The time between the declaration of an SF or SD condition
           and the initialization of the protection switching
           algorithm.";
        reference "RFC4427";
      }
      leaf wait-to-revert {
        type uint16;
        units seconds;
        description
         "Time to wait before attempting LSP reversion";
        reference "RFC4427";
      }
      leaf aps-signal-id {
        type uint8 {
          range "1..255";
        }
        default 1;
        description
          "The APS signal number used to reference the traffic of this
           tunnel. The default value for normal traffic is 1.
           The default value for extra-traffic is 255. If not specified,
```

```
            non-default values can be assigned by the server,
            if and only if, the server controls both endpoints.";
          reference "RFC4427";
        }
      }
      container restoration {
        description "Restoration parameters";
        leaf enable {
          type boolean;
          default 'false';
          description
            "A flag to specify if LSP restoration is enabled";
          reference "RFC4427";
        }
        leaf restoration-type {
          type identityref {
            base te-types:lsp-restoration-type;
          }
          default te-types:lsp-restoration-restore-any;
          description "LSP restoration type.";
        }
        leaf restoration-scheme {
          type identityref {
            base te-types:restoration-scheme-type;
          }
          default te-types:restoration-scheme-preconfigured;
          description "LSP restoration scheme.";
        }
        leaf restoration-reversion-disable {
          type boolean;
          default 'false';
          description "Disable restoration reversion to working path";
        }
        leaf hold-off-time {
          type uint32;
          units "milli-seconds";
          description
            "The time between the declaration of an SF or SD condition
             and the initialization of the protection switching
             algorithm.";
          reference "RFC4427";
        }
        leaf wait-to-restore {
          type uint16;
          units seconds;
          description
            "Time to wait before attempting LSP restoration";
          reference "RFC4427";
```

```
      }
      leaf wait-to-revert {
        type uint16;
        units seconds;
        description
          "Time to wait before attempting LSP reversion";
        reference "RFC4427";
      }
    }
  }

  grouping p2p-dependency-tunnels-properties {
    description
      "Groupong for tunnel dependency list of tunnels";
    container dependency-tunnels {
      description "Dependency tunnels list";
      list dependency-tunnel {
        key "name";
        description "Dependency tunnel entry";
        leaf name {
          type leafref {
            path "../../../../../tunnels/tunnel/name";
            require-instance 'false';
          }
          description "Dependency tunnel name";
        }
        leaf encoding {
          type identityref {
            base te-types:lsp-encoding-types;
          }
          default te-types:lsp-encoding-packet;
          description "LSP encoding type";
          reference "RFC3945";
        }
        leaf switching-type {
          type identityref {
            base te-types:switching-capabilities;
          }
          default te-types:switching-psc1;
          description "LSP switching type";
          reference "RFC3945";
        }
      }
    }
  }

  grouping tunnel-p2p-config {
    description
```

```
      "Configuration parameters relating to TE tunnel";
    leaf name {
      type string;
      description "TE tunnel name.";
    }
    leaf identifier {
      type uint16;
      description
        "TE tunnel Identifier.";
      reference "RFC3209";
    }
    leaf description {
      type string;
      default 'None';
      description
        "Textual description for this TE tunnel";
    }
    leaf encoding {
      type identityref {
        base te-types:lsp-encoding-types;
      }
      default te-types:lsp-encoding-packet;
      description "LSP encoding type";
      reference "RFC3945";
    }
    leaf switching-type {
      type identityref {
        base te-types:switching-capabilities;
      }
      default te-types:switching-psc1;
      description "LSP switching type";
      reference "RFC3945";
    }
    leaf provisioning-state {
      type identityref {
        base te-types:tunnel-state-type;
      }
      default te-types:tunnel-state-up;
      description "TE tunnel administrative state.";
    }
    leaf preference {
      type uint8 {
        range "1..255";
      }
      default 100;
      description
        "Specifies a preference for this tunnel.
         A lower number signifies a better preference";
```

```
    }
    leaf reoptimize-timer {
      type uint16;
      units seconds;
      description
       "frequency of reoptimization of a traffic engineered LSP";
    }
    leaf source {
      type te-types:te-node-id;
      description "TE tunnel source node ID.";
    }
    leaf destination {
      type te-types:te-node-id;
      description "TE tunnel destination node ID";
    }
    leaf src-tp-id {
      type yang:hex-string;
      default '00:00:00:00';
      description
        "TE tunnel source termination point identifier.";
    }
    leaf dst-tp-id {
      type yang:hex-string;
      default '00:00:00:00';
      description
        "TE tunnel destination termination point identifier.";
    }
    leaf bidirectional {
      type boolean;
      default 'false';
      description "TE tunnel bidirectional";
    }
    uses tunnel-p2p-associations-properties;
    uses protection-restoration-properties;
    uses te-types:tunnel-constraints;
    uses p2p-dependency-tunnels-properties;
    uses hierarchical-link-properties;
  }

  grouping tunnel-p2p-associations-properties {
    description "TE tunnel association grouping";
    container association-objects {
      description "TE tunnel associations";
      list association-object {
        key "type ID source global-source";
        description "List of association base objects";
        reference "RFC4872";
        leaf type {
```

```
          type identityref {
            base te-types:association-type;
          }
          description "Association type";
          reference "RFC4872";
        }
        leaf ID {
          type uint16;
          description "Association ID";
          reference "RFC4872";
        }
        leaf source {
          type te-types:te-node-id;
          description "Association source";
          reference "RFC4872";
        }
        leaf global-source {
          type te-types:te-node-id;
          description "Association global source";
          reference "RFC4872";
        }
      }
      list association-object-extended {
        key "type ID source global-source extended-ID";
        description "List of extended association objects";
        reference "RFC6780";
        leaf type {
          type identityref {
            base te-types:association-type;
          }
          description "Association type";
        }
        leaf ID {
          type uint16;
          description "Association ID";
          reference "RFC4872";
        }
        leaf source {
          type te-types:te-node-id;
          description "Association source";
        }
        leaf global-source {
          type te-types:te-node-id;
          description "Association global source";
          reference "RFC4872";
        }
        leaf extended-ID {
          type yang:hex-string;
```

```
          description "Association extended ID";
          reference "RFC4872";
        }
      }
    }
  }

  grouping path-access-segment-info {
    description
      "If an end-to-end tunnel crosses multiple domains using
       the same technology, some additional constraints have to be
       taken in consideration in each domain";
    container path-in-segment {
      presence
        "The end-to-end tunnel starts in a previous domain;
         this tunnel is a segment in the current domain.";
      description
        "This tunnel is a segment that needs to be coordinated
         with previous segment stitched on head-end side.";
      uses te-types:label-set-info;
    }
    container path-out-segment {
      presence
        "The end-to-end tunnel is not terminated in this domain;
         this tunnel is a segment in the current domain.";
      description
        "This tunnel is a segment that needs to be coordinated
         with previous segment stitched on head-end side.";
      uses te-types:label-set-info;
    }
  }

  /* TE tunnel configuration/state grouping */
  grouping tunnel-p2mp-properties {
    description
      "Top level grouping for P2MP tunnel properties.";
    leaf name {
      type string;
      description "TE tunnel name.";
    }
    leaf identifier {
      type uint16;
      description
        "TE tunnel Identifier.";
      reference "RFC3209";
    }
    leaf description {
      type string;
```

```
      default 'None';
      description
        "Textual description for this TE tunnel";
    }
    leaf operational-state {
      type identityref {
        base te-types:tunnel-state-type;
      }
      default te-types:tunnel-state-up;
      config 'false';
      description "TE tunnel administrative state.";
    }
  }

  grouping p2p-path-candidate-secondary-path-config {
    description
      "Configuration parameters relating to a secondary path which
      is a candidate for a particular primary path";

    leaf secondary-path {
      type leafref {
        path "../../../../../p2p-secondary-paths/" +
             "p2p-secondary-path/name";
      }
      description
        "A reference to the secondary path that should be utilised
        when the containing primary path option is in use";
    }

    leaf path-setup-protocol {
      type identityref {
        base te-types:path-signaling-type;
      }
      default te-types:path-setup-static;
      description
        "Signaling protocol used to set up this tunnel";
    }
  }

  grouping p2p-secondary-reverse-path-properties {
    description
      "Configuration parameters relating to a secondary path which
       is a candidate for a particular primary path";

    leaf secondary-path {
      type leafref {
        path "../../../../../p2p-secondary-paths/" +
             "p2p-secondary-path/name";
```

```
      }
      description
        "A reference to the secondary path that should be utilised
        when the containing primary path option is in use";
    }

    leaf path-setup-protocol {
      type identityref {
        base te-types:path-signaling-type;
      }
      default te-types:path-setup-static;
      description
        "Signaling protocol used to set up this tunnel";
    }
  }

  grouping tunnel-p2p-properties {
    description
      "Top level grouping for tunnel properties.";
    leaf operational-state {
      type identityref {
        base te-types:tunnel-state-type;
      }
      default te-types:tunnel-state-up;
      config 'false';
      description "TE tunnel administrative state.";
    }
    uses tunnel-p2p-config;
    container p2p-primary-paths {
      description "Set of P2P primary aths container";
      list p2p-primary-path {
        key "name";
        description
          "List of primary paths for this tunnel.";
        uses p2p-primary-path-properties;
        uses p2p-primary-reverse-path-properties;
        container candidate-p2p-secondary-paths {
          description
            "The set of candidate secondary paths which may be used
            for this primary path. When secondary paths are specified
            in the list the path of the secondary LSP in use must be
            restricted to those path options referenced. The
            priority of the secondary paths is specified within the
            list. Higher priority values are less preferred - that is
            to say that a path with priority 0 is the most preferred
            path. In the case that the list is empty, any secondary
            path option may be utilised when the current primary path
            is in use.";
```

```
          list candidate-p2p-secondary-path {
            key "secondary-path";
            description
              "List of secondary paths for this tunnel.";
            uses p2p-path-candidate-secondary-path-config;

            leaf active {
              type boolean;
              config 'false';
              description
                "Indicates the current active path option that has
                been selected of the candidate secondary paths";
            }
          }
        }
      }
    }
    container p2p-secondary-paths {
      description "Set of P2P secondary paths container";
      list p2p-secondary-path {
        key "name";
        description
          "List of secondary paths for this tunnel.";
        uses p2p-secondary-path-properties;
      }
    }
  }

  grouping shared-resources-tunnels-state {
    description
      "The specific tunnel that is using the shared secondary path
       resources";
    leaf lsp-shared-resources-tunnel {
      type tunnel-ref;
      description
        "Reference to the tunnel that sharing secondary path
        resources with this tunnel";
    }
  }
  grouping shared-resources-tunnels {
    description
      "Set of tunnels that share secondary path resources with
      this tunnnel";
    container shared-resources-tunnels {
      description
        "Set of tunnels that share secondary path resources with
        this tunnnel";
      leaf-list lsp-shared-resources-tunnel {
```

```
        type tunnel-ref;
        description
          "Reference to the tunnel that sharing secondary path
          resources with this tunnel";
      }
    }
  }

  grouping tunnel-actions {
    description "Tunnel actions";
    action tunnel-action {
      description "Tunnel action";
      input {
        leaf action-type {
          type identityref {
            base te-types:tunnel-action-type;
          }
          description "Tunnel action type";
        }
      }
      output {
        leaf action-result {
          type identityref {
            base te-types:te-action-result;
          }
          description "The result of the RPC operation";
        }
      }
    }
  }
  grouping tunnel-protection-actions {
    description
      "Protection external command actions";
    action protection-external-commands {
      input {
        leaf protection-external-command {
          type identityref {
            base te-types:protection-external-commands;
          }
          description
            "Protection external command";
        }
        leaf protection-group-ingress-node-id {
          type te-types:te-node-id;
          description
            "When specified, indicates whether the action is
             applied on ingress node.
             By default, if neither ingress nor egress node-id
```

```
            is set, the the action applies to ingress node only.";
        }
        leaf protection-group-egress-node-id {
          type te-types:te-node-id;
          description
            "When specified, indicates whether the action is
             applied on egress node.
             By default, if neither ingress nor egress node-id
             is set, the the action applies to ingress node only.";
        }
        leaf path-ref {
          type path-ref;
          description
            "Indicates to which path the external command applies to.";
        }
        leaf traffic-type {
          type enumeration {
            enum normal-traffic {
              description
                "The manual-switch or forced-switch command applies to
                 the normal traffic (this Tunnel).";
            }
            enum null-traffic {
              description
                "The manual-switch or forced-switch command applies to
                 the null traffic.";
            }
            enum extra-traffic {
              description
                "The manual-switch or forced-switch command applies to
                 the extra traffic (the extra-traffic Tunnel sharing
                 protection bandwidth with this Tunnel).";
            }
          }
          description
            "Indicates whether the manual-switch or forced-switch
             commands applies to the normal traffic, the null traffic
             or the extra-traffic.";
          reference "RFC4427";
        }
        leaf extra-traffic-tunnel-ref {
          type tunnel-ref;
          description
            "In case there are multiple extra-traffic tunnels sharing
             protection bandwidth with this Tunnel (m:n protection),
             represents which extra-traffic Tunnel the manual-switch or
             forced-switch to extra-traffic command applies to.";
        }
```

```
      }
    }
  }

  /*** End of TE tunnel groupings ***/

  /**
   * LSP related generic groupings
   */
  grouping lsp-record-route-information-state {
    description "recorded route information grouping";
    container lsp-record-route-information {
      description "RSVP recorded route object information";
      list lsp-record-route-information {
        when "../../origin-type = 'ingress'" {
          description "Applicable on ingress LSPs only";
        }
        key "index";
        description "Record route list entry";
        uses te-types:record-route-state;
      }
    }
  }

  grouping lsps-state-grouping {
    description
      "LSPs state operational data grouping";
    container lsps-state {
      config 'false';
      description "TE LSPs state container";
      list lsp {
        key
          "source destination tunnel-id lsp-id "+
          "extended-tunnel-id";
        description "List of LSPs associated with the tunnel.";
        uses lsp-properties-state;
        uses lsp-record-route-information-state;
      }
    }
  }

  /*** End of TE LSP groupings ***/

  /**
   * TE global generic groupings
   */

  /* Global named admin-groups configuration data */
```

```
  grouping named-admin-groups-properties {
    description
      "Global named administrative groups configuration
      grouping";
    leaf name {
      type string;
      description
        "A string name that uniquely identifies a TE
        interface named admin-group";
    }
    leaf bit-position {
      type uint32;
      description
        "Bit position representing the administrative group";
      reference "RFC3209 and RFC7308";
    }
  }
  grouping named-admin-groups {
    description
      "Global named administrative groups configuration
      grouping";
    container named-admin-groups {
      description "TE named admin groups container";
      list named-admin-group {
        if-feature te-types:extended-admin-groups;
        if-feature te-types:named-extended-admin-groups;
        key "name";
        description
          "List of named TE admin-groups";
        uses named-admin-groups-properties;
      }
    }
  }

  /* Global named admin-srlgs configuration data */
  grouping named-srlgs-properties {
    description
      "Global named SRLGs configuration grouping";
    leaf name {
      type string;
      description
        "A string name that uniquely identifies a TE
        interface named srlg";
    }
    leaf group {
      type te-types:srlg;
      description "An SRLG value";
    }
```

```
    leaf cost {
      type uint32;
      description
        "SRLG associated cost. Used during path to append
         the path cost when traversing a link with this SRLG";
    }
  }

  grouping named-srlgs {
    description
      "Global named SRLGs configuration grouping";
    container named-srlgs {
      description "TE named SRLGs container";
      list named-srlg {
        if-feature te-types:named-srlg-groups;
        key "name";
        description
          "A list of named SRLG groups";
        uses named-srlgs-properties;
      }
    }
  }

  /* Global named paths constraints configuration data */
  grouping path-constraints-state {
    description "TE path constraints state";
    leaf bandwidth {
      type te-types:te-bandwidth;
      config 'false';
      description
        "A technology agnostic requested bandwidth to use
         for path computation";
    }
    leaf disjointness-type {
      type te-types:te-path-disjointness;
      config 'false';
      description
        "The type of resource disjointness.";
    }
  }

  grouping path-constraints-common {
    description
      "Global named path constraints configuration
       grouping";
    uses te-types:common-path-constraints-attributes;
    uses te-types:generic-path-disjointness;
    uses te-types:path-constraints-route-objects;
```

```
    uses shared-resources-tunnels {
      description
        "Set of tunnels that are allowed to share secondary path
         resources of this tunnel";
    }
    uses path-access-segment-info {
      description
        "Tunnel constraints induced by other segments.";
    }
  }

  grouping named-path-constraints {
    description
      "Global named path constraints configuration
      grouping";
    container named-path-constraints {
      description "TE named path constraints container";
      list named-path-constraint {
        if-feature te-types:named-path-constraints;
        key "name";
        leaf name {
          type string;
          description
            "A string name that uniquely identifies a
            path constraint set";
        }
        uses path-constraints-common;
        description
          "A list of named path constraints";
      }
    }
  }

  /* TE globals container data */
  grouping globals-grouping {
    description
      "Globals TE system-wide configuration data grouping";
    container globals {
      description
        "Globals TE system-wide configuration data container";
      uses named-admin-groups;
      uses named-srlgs;
      uses named-path-constraints;
    }
  }

  /* TE tunnels container data */
  grouping tunnels-grouping {
```

```
    description
      "Tunnels TE configuration data grouping";
    container tunnels {
      description
        "Tunnels TE configuration data container";

      list tunnel {
        key "name";
        description "P2P TE tunnels list.";
        uses tunnel-p2p-properties;
        uses tunnel-actions;
        uses tunnel-protection-actions;
      }
      list tunnel-p2mp {
        key "name";
        unique "identifier";
        description "P2MP TE tunnels list.";
        uses tunnel-p2mp-properties;
      }
    }
  }

  /* TE LSPs ephemeral state container data */
  grouping lsp-properties-state {
    description
      "LSPs state operational data grouping";
    leaf source {
      type te-types:te-node-id;
      description
        "Tunnel sender address extracted from
        SENDER_TEMPLATE  object";
      reference "RFC3209";
    }
    leaf destination {
      type te-types:te-node-id;
      description
        "Tunnel endpoint address extracted from
        SESSION object";
      reference "RFC3209";
    }
    leaf tunnel-id {
      type uint16;
      description
        "Tunnel identifier used in the SESSION
        that remains constant over the life
        of the tunnel.";
      reference "RFC3209";
    }
```

```
    leaf lsp-id {
      type uint16;
      description
        "Identifier used in the SENDER_TEMPLATE
        and the FILTER_SPEC that can be changed
        to allow a sender to share resources with
        itself.";
      reference "RFC3209";
    }
    leaf extended-tunnel-id {
      type yang:dotted-quad;
       description
        "Extended Tunnel ID of the LSP.";
      reference "RFC3209";
    }
    leaf operational-state {
      type identityref {
        base te-types:lsp-state-type;
      }
      description "LSP operational state.";
    }
    leaf path-setup-protocol {
      type identityref {
        base te-types:path-signaling-type;
      }
      default te-types:path-setup-static;
      description
        "Signaling protocol used to set up this tunnel";
    }
    leaf origin-type {
      type enumeration {
        enum ingress {
          description
            "Origin ingress";
        }
        enum egress {
          description
            "Origin egress";
        }
        enum transit {
          description
            "transit";
        }
      }
      default 'ingress';
      description
        "Origin type of LSP relative to the location
        of the local switch in the path.";
```

```
    }

    leaf lsp-resource-status {
      type enumeration {
        enum primary {
          description
            "A primary LSP is a fully established LSP for
             which the resource allocation has been committed
             at the data plane";
        }
        enum secondary {
          description
            "A secondary LSP is an LSP that has been provisioned
             in the control plane only; e.g. resource allocation
             has not been committed at the data plane";
        }
      }
      default 'primary';
      description "LSP resource allocation type";
      reference "RFC4872, section 4.2.1";
    }

    uses protection-restoration-properties-state;
  }
  /*** End of TE global groupings ***/

  /**
   * TE configurations container
   */
  container te {
    presence "Enable TE feature.";
    description
       "TE global container.";

    /* TE Global Configuration Data */
    uses globals-grouping;

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

    /* TE LSPs State Data */
    uses lsps-state-grouping;

  }

  /* TE Global RPCs/execution Data */
  rpc globals-rpc {
    description
```

```
      "Execution data for TE global.";
  }

  /* TE interfaces RPCs/execution Data */
  rpc interfaces-rpc {
    description
      "Execution data for TE interfaces.";
  }

  /* TE Tunnel RPCs/execution Data */
  rpc tunnels-rpc {
    description "TE tunnels RPC nodes";
    input {
      container tunnel-info {
        description "Tunnel Identification";
        choice type {
          description "Tunnel information type";
          case tunnel-p2p {
            leaf p2p-id {
              type tunnel-ref;
              description "P2P TE tunnel";
            }
          }
          case tunnel-p2mp {
            leaf p2mp-id {
              type tunnel-p2mp-ref;
              description "P2MP TE tunnel";
            }
          }
        }
      }
    }
    output {
      container result {
        description
          "The container result of the RPC operation";
        leaf result {
          type enumeration {
            enum success {
              description "Origin ingress";
            }
            enum in-progress {
              description "Origin egress";
            }
            enum fail {
              description "transit";
            }
          }
```

```
          description "The result of the RPC operation";
        }
      }
    }
  }

  /* TE Global Notification Data */
  notification globals-notif {
    description
      "Notification messages for Global TE.";
  }

  /* TE Tunnel Notification Data */
  notification tunnels-notif {
    description
      "Notification messages for TE tunnels.";
  }
}
<CODE ENDS>
```

Figure 7: TE generic YANG module

The TE device YANG module "ietf-te-device" imports the following module(s):

- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-interfaces defined in [RFC8343]
- ietf-routing-types defined in [RFC8294]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]
- ietf-te defined in this document

```
<CODE BEGINS> file "ietf-te-device@2019-04-09.yang"
module ietf-te-device {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-te-device";

  /* Replace with IANA when assigned */
  prefix "te-dev";

  /* Import TE generic types */
  import ietf-te {
    prefix te;
    reference "draft-ietf-teas-yang-te: A YANG Data Model for Traffic
```

```
                Engineering Tunnels and Interfaces";
    }

    /* Import TE generic types */
    import ietf-te-types {
      prefix te-types;
      reference "draft-ietf-teas-yang-te-types: A YANG Data Model for
                 Common Traffic Engineering Types";
    }

    import ietf-interfaces {
      prefix if;
      reference "RFC8343: A YANG Data Model for Interface Management";
    }

    import ietf-inet-types {
      prefix inet;
      reference "RFC6991: Common YANG Data Types";
    }

    import ietf-routing-types {
      prefix "rt-types";
      reference "RFC8294: Common YANG Data Types for the Routing Area";
    }

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

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

       WG Chair: Lou Berger
                 <mailto:lberger@labn.net>

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

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

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

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

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

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

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

      description
        "YANG data module for TE device configurations,
         state, RPC and notifications.";

      revision "2019-04-09" {
        description "Latest update to TE device YANG module.";
        reference
          "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels
           and Interfaces";
      }

      /**
       * TE LSP device state grouping
       */
      grouping lsps-device-state {
        description "TE LSP device state grouping";
        container lsp-timers {
          when "../te:origin-type = 'ingress'" {
            description "Applicable to ingress LSPs only";
          }
          description "Ingress LSP timers";
          leaf life-time {
            type uint32;
            units seconds;
            description
              "lsp life time";
          }

          leaf time-to-install {
            type uint32;
            units seconds;
            description
              "lsp installation delay time";
          }

          leaf time-to-destroy {
            type uint32;
            units seconds;
            description
```

```
            "lsp expiration delay time";
        }
      }

      container downstream-info {
        when "../te:origin-type != 'egress'" {
          description "Applicable to ingress LSPs only";
        }
        description
          "downstream information";

        leaf nhop {
          type inet:ip-address;
          description
            "downstream nexthop.";
        }

        leaf outgoing-interface {
          type if:interface-ref;
          description
            "downstream interface.";
        }

        leaf neighbor {
          type inet:ip-address;
          description
            "downstream neighbor.";
        }

        leaf label {
          type rt-types:generalized-label;
          description
            "downstream label.";
        }
      }

      container upstream-info {
        when "../te:origin-type != 'ingress'" {
          description "Applicable to non-ingress LSPs only";
        }
        description
          "upstream information";

        leaf phop {
          type inet:ip-address;
          description
            "upstream nexthop or previous-hop.";
        }
```

```
        leaf neighbor {
          type inet:ip-address;
          description
            "upstream neighbor.";
        }

        leaf label {
          type rt-types:generalized-label;
          description
            "upstream label.";
        }
      }
    }

    /**
     * Device general groupings.
     */
    grouping tunnel-device-config {
      description "Device TE tunnel configs";
      leaf path-invalidation-action {
        type identityref {
          base te-types:path-invalidation-action-type;
        }
        description "Tunnel path invalidtion action";
      }
    }

    grouping lsp-device-timers-config {
      description "Device TE LSP timers configs";
      leaf lsp-install-interval {
        type uint32;
        units seconds;
        description
          "lsp installation delay time";
      }
      leaf lsp-cleanup-interval {
        type uint32;
        units seconds;
        description
          "lsp cleanup delay time";
      }
      leaf lsp-invalidation-interval {
        type uint32;
        units seconds;
        description
          "lsp path invalidation before taking action delay time";
      }
    }
```

```
     /**
      * TE global device generic groupings
      */

     /* TE interface container data */
     grouping interfaces-grouping {
       description
         "Interface TE configuration data grouping";
       container interfaces {
         description
           "Configuration data model for TE interfaces.";
         uses te-all-attributes;
         list interface {
           key "interface";
           description "TE interfaces.";
           leaf interface {
             type if:interface-ref;
             description
               "TE interface name.";
           }
           /* TE interface parameters */
           uses te-attributes;
         }
       }
     }

     /**
      * TE interface device generic groupings
      */
     grouping te-admin-groups-config {
       description
         "TE interface affinities grouping";
       choice admin-group-type {
         description
           "TE interface administrative groups
           representation type";
         case value-admin-groups {
           choice value-admin-group-type {
             description "choice of admin-groups";
             case admin-groups {
               description
                 "Administrative group/Resource
                 class/Color.";
               leaf admin-group {
                 type te-types:admin-group;
                 description
                   "TE interface administrative group";
               }
```

```
            }
            case extended-admin-groups {
              if-feature te-types:extended-admin-groups;
              description
                "Extended administrative group/Resource
                class/Color.";
              leaf extended-admin-group {
                type te-types:extended-admin-group;
                description
                  "TE interface extended administrativei
                  group";
              }
            }
          }
        }
        case named-admin-groups {
          list named-admin-groups {
            if-feature te-types:extended-admin-groups;
            if-feature te-types:named-extended-admin-groups;
            key named-admin-group;
            description
              "A list of named admin-group entries";
            leaf named-admin-group {
              type leafref {
                path "../../../../te:globals/" +
                  "te:named-admin-groups/te:named-admin-group/" +
                  "te:name";
              }
              description "A named admin-group entry";
            }
          }
        }
      }
    }

    /* TE interface SRLGs */
    grouping te-srlgs-config {
      description "TE interface SRLG grouping";
      choice srlg-type {
        description "Choice of SRLG configuration";
        case value-srlgs {
          list values {
            key "value";
            description "List of SRLG values that
              this link is part of.";
            leaf value {
              type uint32 {
                range "0..4294967295";
```

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

    grouping te-igp-flooding-bandwidth-config {
      description
        "Configurable items for igp flooding bandwidth
        threshold configuration.";
      leaf threshold-type {
        type enumeration {
          enum DELTA {
            description
              "DELTA indicates that the local
              system should flood IGP updates when a
              change in reserved bandwidth >= the specified
              delta occurs on the interface.";
          }
          enum THRESHOLD_CROSSED {
            description
              "THRESHOLD-CROSSED indicates that
              the local system should trigger an update (and
              hence flood) the reserved bandwidth when the
              reserved bandwidth changes such that it crosses,
              or becomes equal to one of the threshold values.";
          }
        }
        description
```

```
          "The type of threshold that should be used to specify the
          values at which bandwidth is flooded. DELTA indicates that
          the local system should flood IGP updates when a change in
          reserved bandwidth >= the specified delta occurs on the
          interface. Where THRESHOLD_CROSSED is specified, the local
          system should trigger an update (and hence flood) the
          reserved bandwidth when the reserved bandwidth changes such
          that it crosses, or becomes equal to one of the threshold
          values";
      }

      leaf delta-percentage {
        when "../threshold-type = 'DELTA'" {
          description
            "The percentage delta can only be specified when the
            threshold type is specified to be a percentage delta of
            the reserved bandwidth";
        }
        type rt-types:percentage;
        description
          "The percentage of the maximum-reservable-bandwidth
          considered as the delta that results in an IGP update
          being flooded";
      }
      leaf threshold-specification {
        when "../threshold-type = 'THRESHOLD_CROSSED'" {
          description
            "The selection of whether mirrored or separate threshold
            values are to be used requires user specified thresholds to
            be set";
        }
        type enumeration {
          enum MIRRORED_UP_DOWN {
            description
              "MIRRORED_UP_DOWN indicates that a single set of
              threshold values should be used for both increasing
              and decreasing bandwidth when determining whether
              to trigger updated bandwidth values to be flooded
              in the IGP TE extensions.";
          }
          enum SEPARATE_UP_DOWN {
            description
              "SEPARATE_UP_DOWN indicates that a separate
              threshold values should be used for the increasing
              and decreasing bandwidth when determining whether
              to trigger updated bandwidth values to be flooded
              in the IGP TE extensions.";
          }
```

```
          }
          description
            "This value specifies whether a single set of threshold
            values should be used for both increasing and decreasing
            bandwidth when determining whether to trigger updated
            bandwidth values to be flooded in the IGP TE extensions.
            MIRRORED-UP-DOWN indicates that a single value (or set of
            values) should be used for both increasing and decreasing
            values, where SEPARATE-UP-DOWN specifies that the increasing
            and decreasing values will be separately specified";
        }

        leaf-list up-thresholds {
          when "../threshold-type = 'THRESHOLD_CROSSED'" +
            "and ../threshold-specification = 'SEPARATE_UP_DOWN'" {
              description
                "A list of up-thresholds can only be specified when the
                bandwidth update is triggered based on crossing a
                threshold and separate up and down thresholds are
                required";
          }
          type rt-types:percentage;
          description
            "The thresholds (expressed as a percentage of the maximum
            reservable bandwidth) at which bandwidth updates are to be
            triggered when the bandwidth is increasing.";
        }

        leaf-list down-thresholds {
          when "../threshold-type = 'THRESHOLD_CROSSED'" +
            "and ../threshold-specification = 'SEPARATE_UP_DOWN'" {
              description
                "A list of down-thresholds can only be specified when the
                bandwidth update is triggered based on crossing a
                threshold and separate up and down thresholds are
                required";
          }
          type rt-types:percentage;
          description
            "The thresholds (expressed as a percentage of the maximum
            reservable bandwidth) at which bandwidth updates are to be
            triggered when the bandwidth is decreasing.";
        }

        leaf-list up-down-thresholds {
          when "../threshold-type = 'THRESHOLD_CROSSED'" +
            "and ../threshold-specification = 'MIRRORED_UP_DOWN'" {
              description
```

```
              "A list of thresholds corresponding to both increasing
              and decreasing bandwidths can be specified only when an
              update is triggered based on crossing a threshold, and
              the same up and down thresholds are required.";
          }
          type rt-types:percentage;
          description
            "The thresholds (expressed as a percentage of the maximum
            reservable bandwidth of the interface) at which bandwidth
            updates are flooded - used both when the bandwidth is
            increasing and decreasing";
        }
      }

      /* TE interface metric */
      grouping te-metric-config {
        description "Interface TE metric grouping";
        leaf te-metric {
          type te-types:te-metric;
          description "Interface TE metric.";
        }
      }

      /* TE interface switching capabilities */
      grouping te-switching-cap-config {
        description
          "TE interface switching capabilities";
        list switching-capabilities {
          key "switching-capability";
          description
            "List of interface capabilities for this interface";
          leaf switching-capability {
            type identityref {
              base te-types:switching-capabilities;
            }
            description
              "Switching Capability for this interface";
          }
          leaf encoding {
            type identityref {
              base te-types:lsp-encoding-types;
            }
            description
              "Encoding supported by this interface";
          }
        }
      }
```

```
    grouping te-advertisements-state {
      description
        "TE interface advertisements state grouping";
      container te-advertisements-state {
        description
          "TE interface advertisements state container";
        leaf flood-interval {
          type uint32;
          description
            "The periodic flooding interval";
        }
        leaf last-flooded-time {
          type uint32;
          units seconds;
          description
            "Time elapsed since last flooding in seconds";
        }
        leaf next-flooded-time {
          type uint32;
          units seconds;
          description
            "Time remained for next flooding in seconds";
        }
        leaf last-flooded-trigger {
          type enumeration {
            enum link-up {
              description "Link-up flooding trigger";
            }
            enum link-down {
              description "Link-up flooding trigger";
            }
            enum threshold-up {
              description
                "Bandwidth reservation up threshold";
            }
            enum threshold-down {
              description
                "Bandwidth reservation down threshold";
            }
            enum bandwidth-change {
              description "Banwidth capacity change";
            }
            enum user-initiated {
              description "Initiated by user";
            }
            enum srlg-change {
              description "SRLG property change";
            }
```

```
            enum periodic-timer {
              description "Periodic timer expired";
            }
          }
          default 'periodic-timer';
          description "Trigger for the last flood";
        }
        list advertized-level-areas {
          key level-area;
          description
            "List of areas the TE interface is advertised
            in";
          leaf level-area {
            type uint32;
            description
              "The IGP area or level where the TE
              interface state is advertised in";
          }
        }
      }
    }

    /* TE interface attributes grouping */
    grouping te-attributes {
      description "TE attributes configuration grouping";
      uses te-metric-config;
      uses te-admin-groups-config;
      uses te-srlgs-config;
      uses te-igp-flooding-bandwidth-config;
      uses te-switching-cap-config;
      container state {
        config false;
        description
          "State parameters for interface TE metric";
        uses te-advertisements-state;
      }
    }

    grouping te-all-attributes {
      description
        "TE attributes configuration grouping for all
         interfaces";
      uses te-igp-flooding-bandwidth-config;
    }
    /*** End of TE interfaces device groupings ***/


    /**
```

```
     * TE device augmentations
     */
    augment "/te:te" {
      description "TE global container.";
      /* TE Interface Configuration Data */
      uses interfaces-grouping;
      container performance-thresholds {
        description
          "Performance parameters configurable thresholds";
      }
    }

    /* TE globals device augmentation */
    augment "/te:te/te:globals" {
      description
        "Global TE device specific configuration parameters";
      uses lsp-device-timers-config;
    }

    /* TE tunnels device configuration augmentation */
    augment "/te:te/te:tunnels/te:tunnel" {
      description
        "Tunnel device dependent augmentation";
      uses lsp-device-timers-config;
    }

    /* TE LSPs device state augmentation */
    augment "/te:te/te:lsps-state/te:lsp" {
       description
         "LSP device dependent augmentation";
       uses lsps-device-state;
    }

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

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

    /* TE interfaces RPCs/execution Data */
```

```
    rpc interfaces-rpc {
      description
        "Execution data for TE interfaces.";
    }

    /* TE Interfaces Notification Data */
    notification interfaces-notif {
      description
        "Notification messages for TE interfaces.";
    }
  }
  <CODE ENDS>
```

Figure 8: TE device specific YANG module

5. IANA Considerations

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

```
   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-device
   XML: N/A, the requested URI is an XML namespace.
```

This document registers two YANG modules in the YANG Module Names registry [RFC6020].

```
   name:       ietf-te
   namespace:  urn:ietf:params:xml:ns:yang:ietf-te
   prefix:     ietf-te
   reference:  RFCXXXX

   name:       ietf-te-device
   namespace:  urn:ietf:params:xml:ns:yang:ietf-te-device
   prefix:     ietf-te-device
   reference:  RFCXXXX
```

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF

users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations. Following are the subtrees and data nodes and their sensitivity/vulnerability:

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

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

"/te/lsps-state": This list specifies the state derived LSPs. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

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

7. Acknowledgement

The authors would like to thank the members of the multi-vendor YANG design team who are involved in the definition of this model.

The authors would also like to thank Loa Andersson, Lou Berger, Sergio Belotti, Italo Busi, Carlo Perocchio, Francesco Lazzeri, Aihua Guo, Dhruv Dhody, Anurag Sharma, and Xian Zhang for their comments and providing valuable feedback on this document.

8. Contributors

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

9. References

9.1. Normative References

[I-D.ietf-teas-yang-path-computation]
Busi, I., Belotti, S., Lopezalvarez, V., Dios, O., Sharma, A., Shi, Y., Vilata, R., Sethuraman, K., Scharf, M., and D. Ceccarelli, "Yang model for requesting Path Computation", draft-ietf-teas-yang-path-computation-05 (work in progress), March 2019.

[I-D.ietf-teas-yang-rsvp]
Beeram, V., Saad, T., Gandhi, R., Liu, X., Bryskin, I., and H. Shah, "A YANG Data Model for Resource Reservation Protocol (RSVP)", draft-ietf-teas-yang-rsvp-10 (work in progress), February 2019.

[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.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <https://www.rfc-editor.org/info/rfc3473>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC3945] Mannie, E., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, DOI 10.17487/RFC3945, October 2004, <https://www.rfc-editor.org/info/rfc3945>.

[RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, DOI 10.17487/RFC4206, October 2005, <https://www.rfc-editor.org/info/rfc4206>.

[RFC4872] Lang, J., Ed., Rekhter, Y., Ed., and D. Papadimitriou, Ed., "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery", RFC 4872, DOI 10.17487/RFC4872, May 2007, <https://www.rfc-editor.org/info/rfc4872>.

[RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <https://www.rfc-editor.org/info/rfc4875>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6107] Shiomoto, K., Ed. and A. Farrel, Ed., "Procedures for Dynamically Signaled Hierarchical Label Switched Paths", RFC 6107, DOI 10.17487/RFC6107, February 2011, <https://www.rfc-editor.org/info/rfc6107>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6780] Berger, L., Le Faucheur, F., and A. Narayanan, "RSVP ASSOCIATION Object Extensions", RFC 6780, DOI 10.17487/RFC6780, October 2012, <https://www.rfc-editor.org/info/rfc6780>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", RFC 7308, DOI 10.17487/RFC7308, July 2014, <https://www.rfc-editor.org/info/rfc7308>.

[RFC7551] Zhang, F., Ed., Jing, R., and R. Gandhi, Ed., "RSVP-TE Extensions for Associated Bidirectional Label Switched Paths (LSPs)", RFC 7551, DOI 10.17487/RFC7551, May 2015, <https://www.rfc-editor.org/info/rfc7551>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

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

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

9.2. Informative References

[RFC4427] Mannie, E., Ed. and D. Papadimitriou, Ed., "Recovery (Protection and Restoration) Terminology for Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4427, DOI 10.17487/RFC4427, March 2006, <https://www.rfc-editor.org/info/rfc4427>.

Authors' Addresses

Tarek Saad
Juniper Networks

Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Igor Bryskin
Huawei Technologies

Email: Igor.Bryskin@huawei.com

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: November 23, 2019

May 23, 2019

YANG Data Model for Traffic Engineering (TE) Topologies
draft-ietf-teas-yang-te-topo-21

Status of this Memo

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

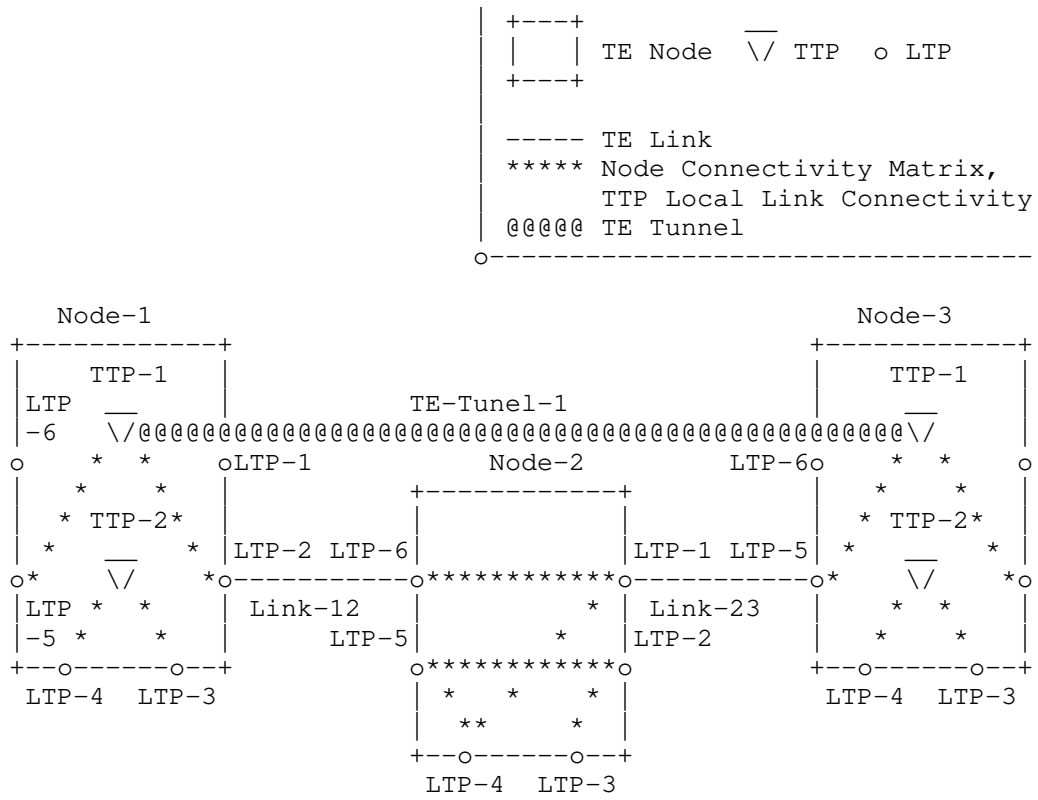


Figure 1: TE Topology Modeling Abstractions

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

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

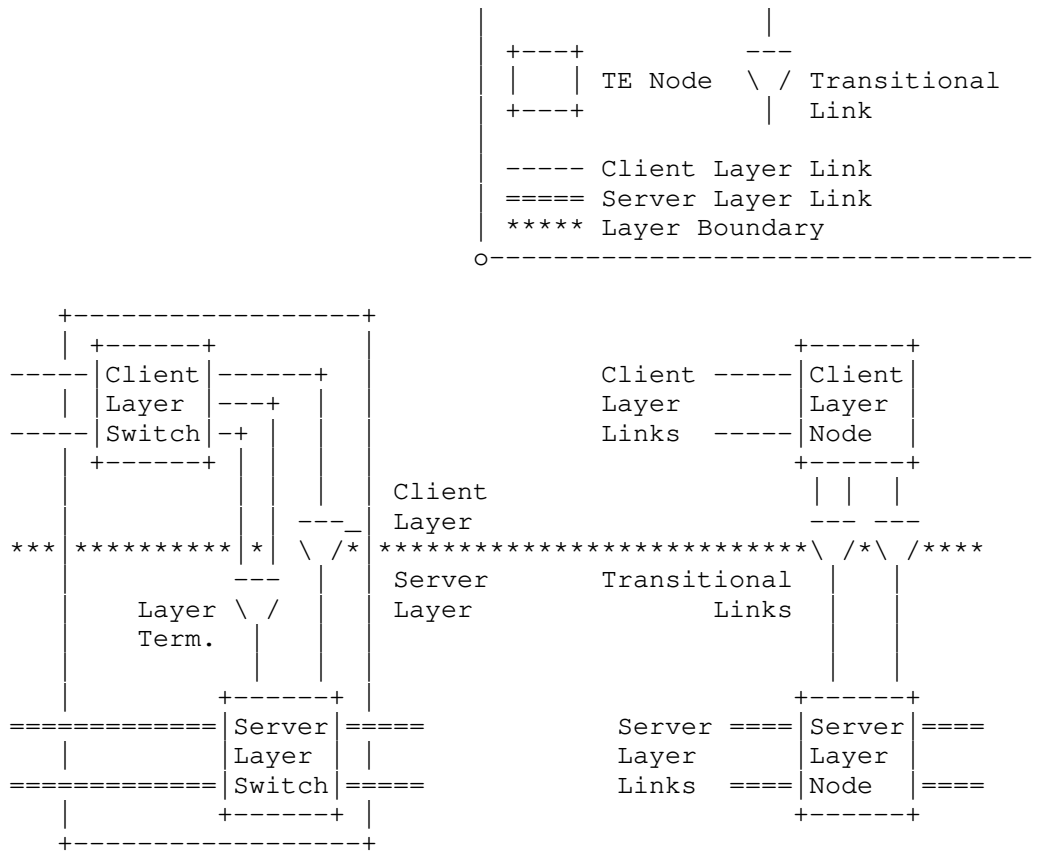


Figure 2: Modeling a Multi-Layer Node (Dual-Layer Example)

3.5. TE Link Termination Point (LTP)

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

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

3.6. TE Tunnel Termination Point (TTP)

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

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

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

3.7. TE Node Connectivity Matrix

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

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

3.8. TTP Local Link Connectivity List (LLCL)

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

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

3.9. TE Path

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

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

3.10. TE Inter-Layer Lock

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

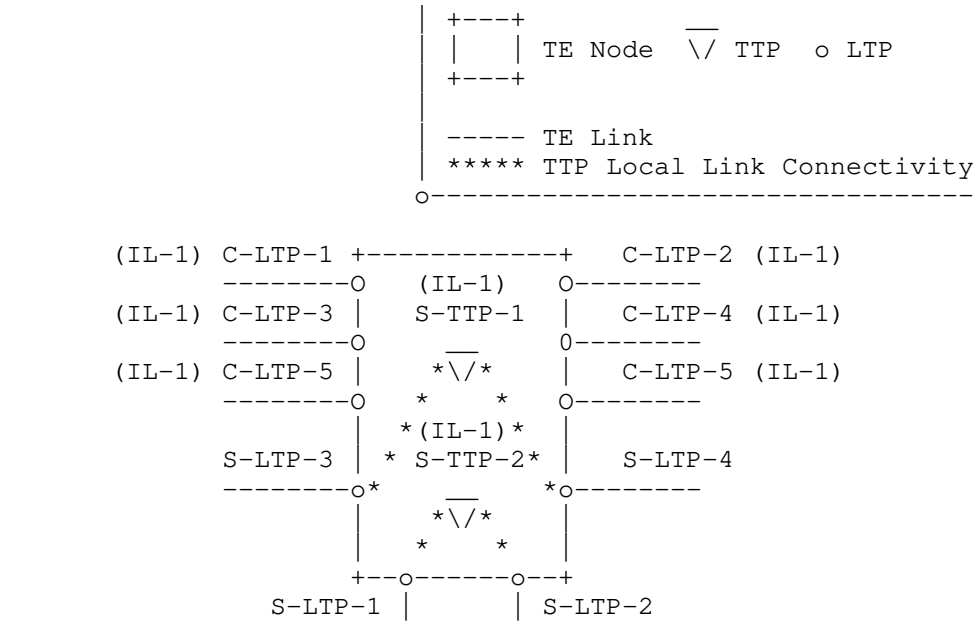


Figure 3: TE Inter-Layer Lock ID Associations

On the picture above a TE inter-layer lock with IL_1 ID associates 6 client layer LTPs (C-LTP-1 - C-LTP-6) with two server layer TTPs (S-TTP-1 and S-TTP-2). They all have the same attribute - TE inter-layer lock ID: IL-1, which is the only thing that indicates the association. A given LTP may have 0, 1 or more inter-layer lock IDs. In the latter case this means that the data arriving at the LTP may be adopted onto any of TTPs associated with all specified inter-layer locks. For example, C-LTP-1 could have two inter-layer lock IDs - IL-1 and IL-2. This would mean that C-LTP-1 for adaptation purposes could use not just TTPs associated with inter-layer lock IL-1 (i.e.

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

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

3.11. Underlay TE topology

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

3.12. Overlay TE topology

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

3.13. Abstract TE topology

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

Figure 4 shows an example of abstract TE topology.

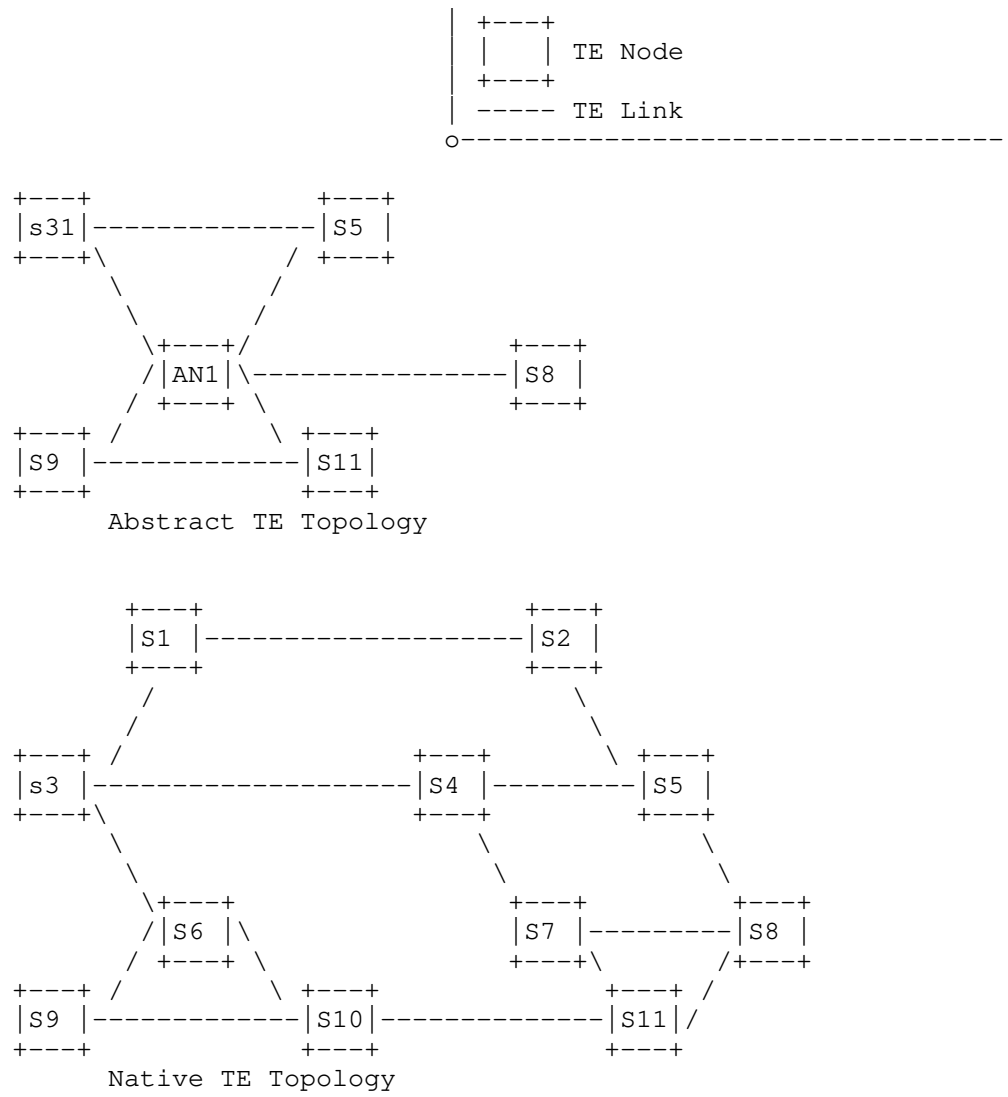


Figure 4: Abstract TE Topology

4. Model Applicability

4.1. Native TE Topologies

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

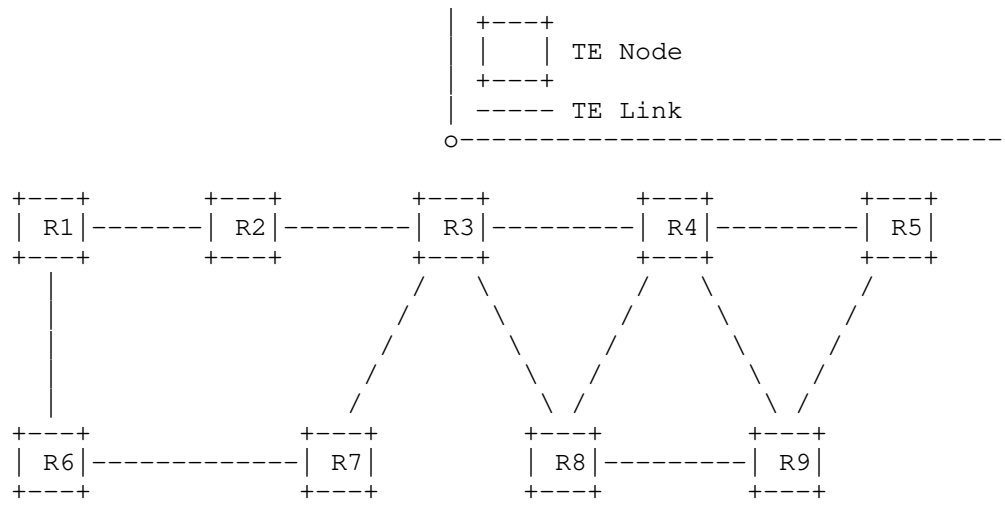


Figure 5a: Example Network Topology

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

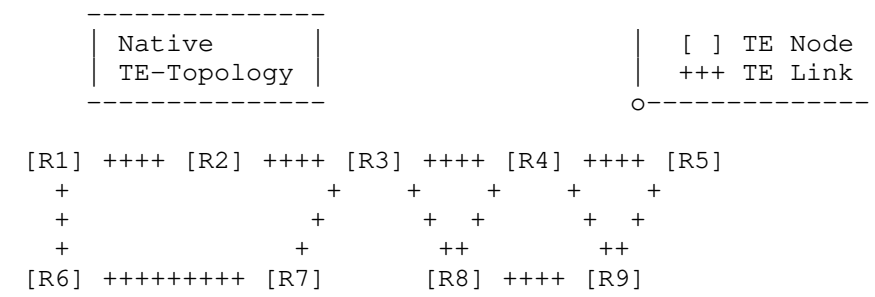


Figure 5b: Native TE Topology as seen on Node R3

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

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

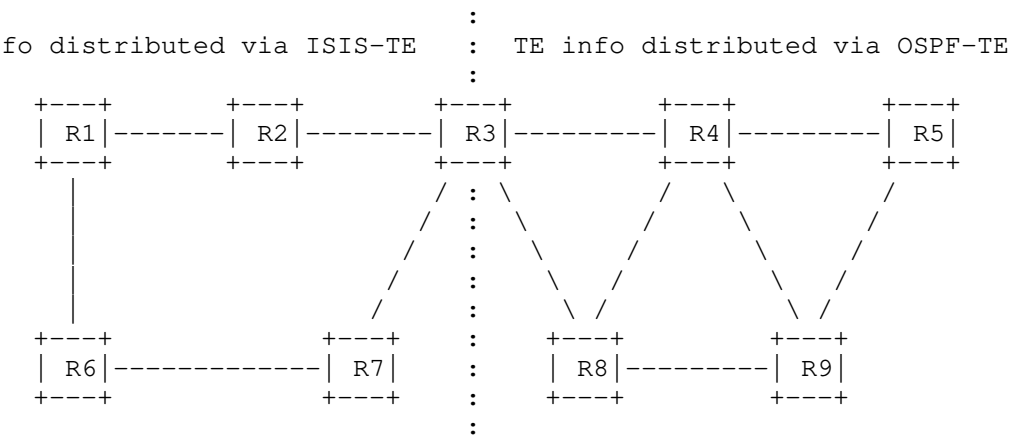


Figure 6a: Example Network Topology

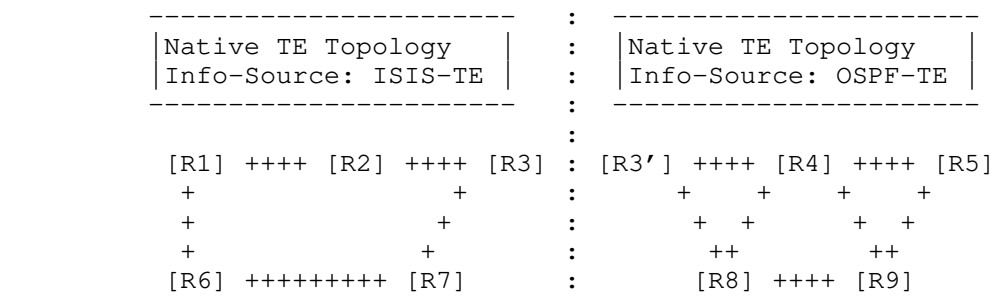


Figure 6b: Native TE Topologies as seen on Node R3

4.2. Customized TE Topologies

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

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

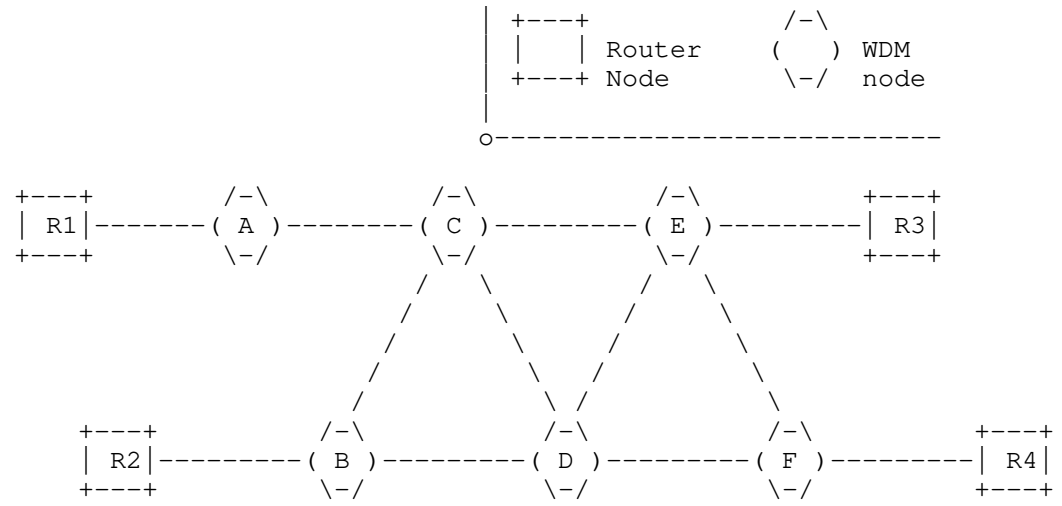


Figure 7: Example packet optical topology

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

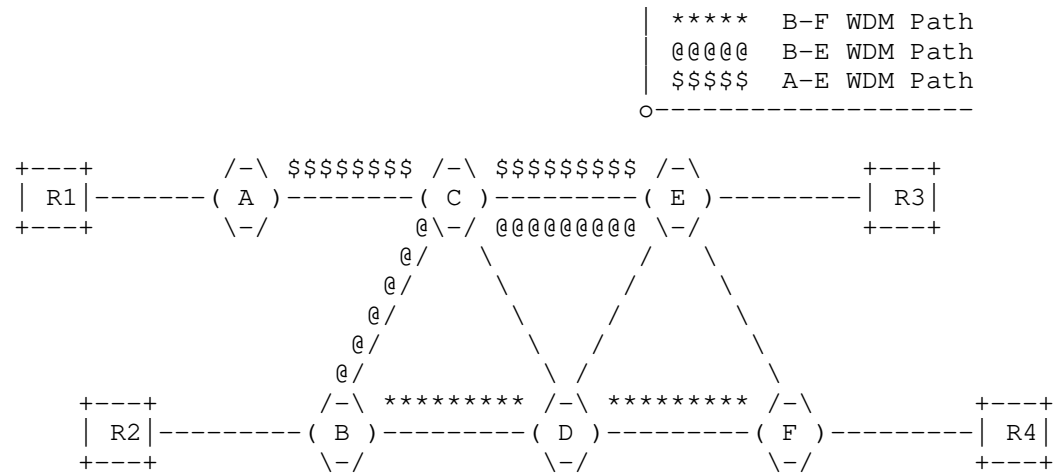


Figure 8a: Paths within the provider domain

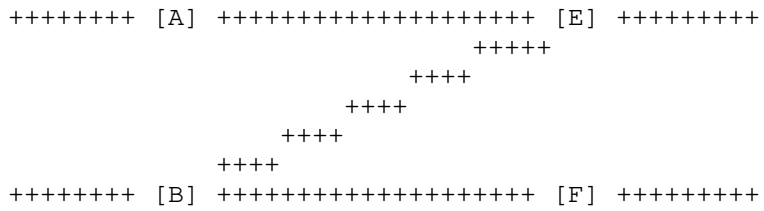


Figure 8b: Customized TE Topology provided to the Client

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.

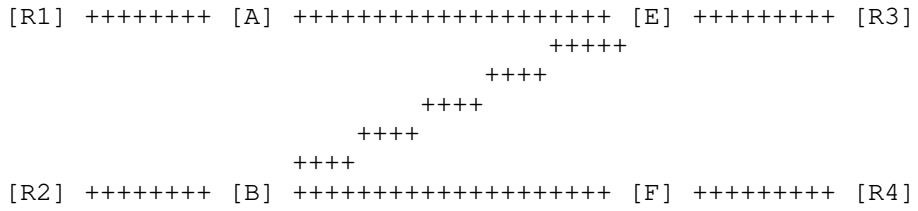


Figure 8c: Customized TE Topology merged with the Client's Native TE Topology

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-connectivit-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 a 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.

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.

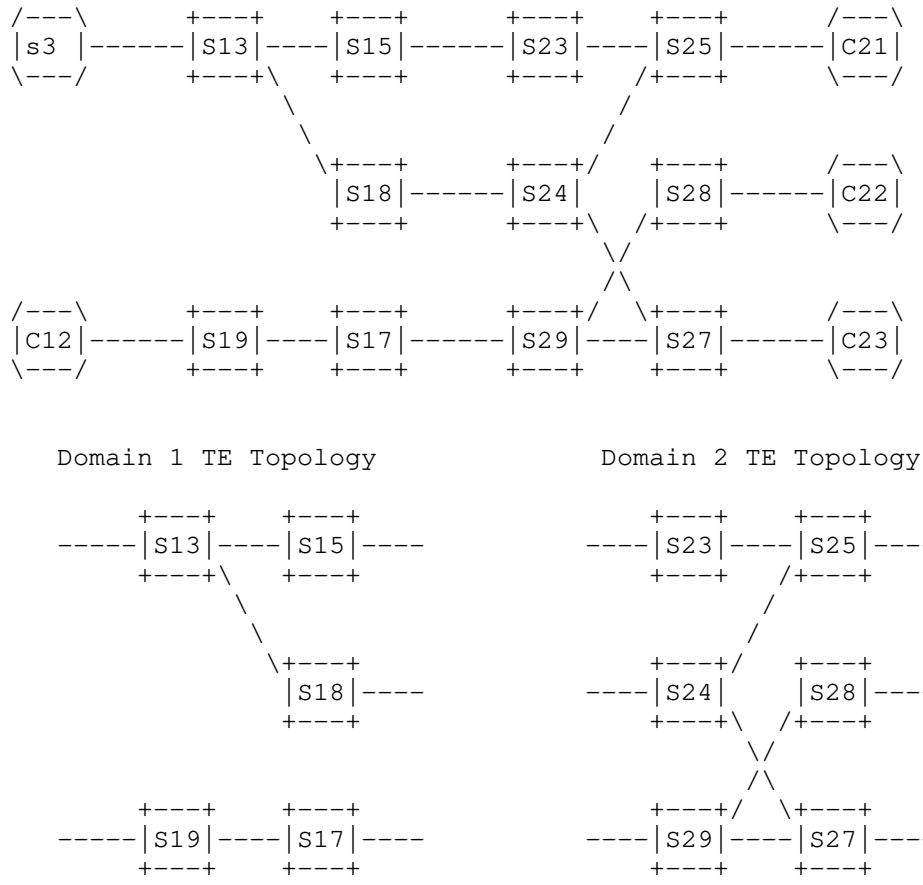


Figure 9: Merging Domain TE Topologies

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

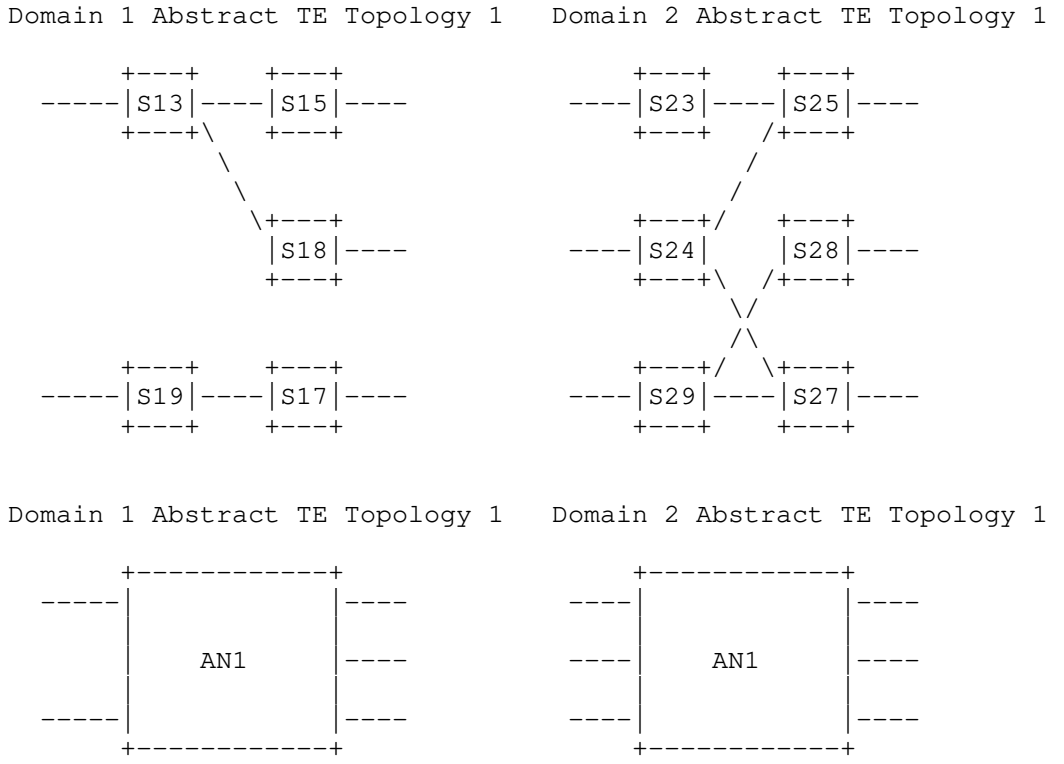


Figure 10: Merging Domain TE Topologies

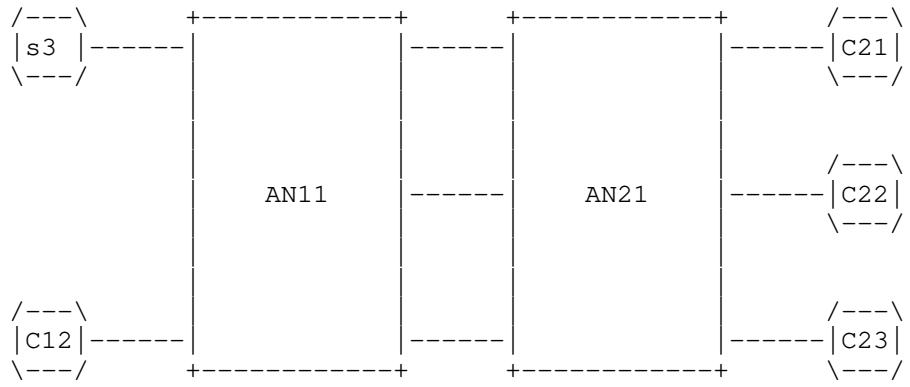
Based on local configuration, templates and/or policies pushed by the client, a given provider may expose more than one abstract TE topology to the client. For example, one abstract TE topology could be optimized based on a lowest-cost criterion, while another one could be based on best possible delay 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



Client's Merged TE Topology 3

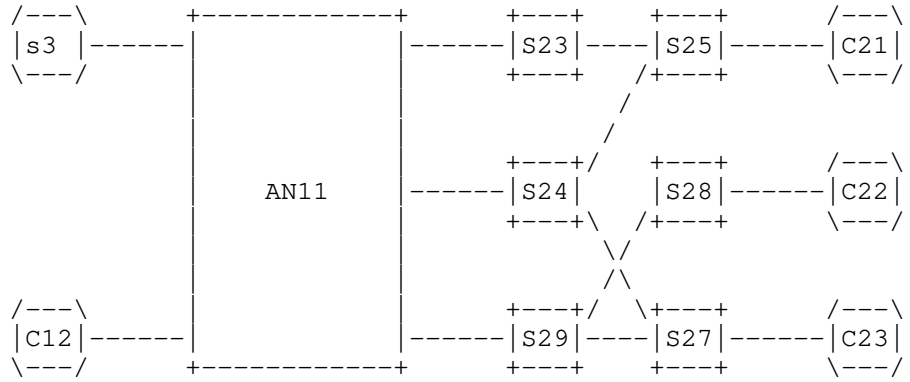


Figure 11: Multiple Native (Merged) Client's TE Topologies

It is important to note that each of the three native (merged) TE topologies could be used by the client for computing TE paths for any of the multi-domain services. The choice as to which topology to use for a given service depends on the service parameters/requirements and the topology's style, optimization criteria and the level of details.

5. Modeling Considerations

5.1. Network topology building blocks

The network topology building blocks are discussed in [RFC8345]. The TE Topology model proposed in this document augments and uses the ietf-network-topology module defined in [RFC8345].

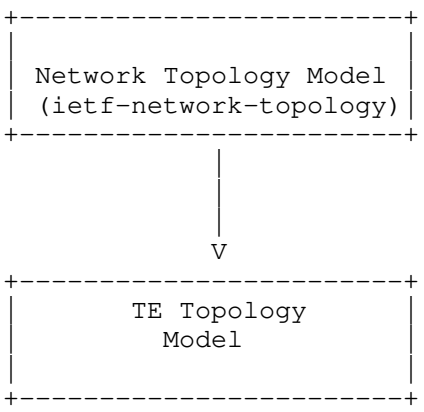


Figure 12: Augmenting the Network Topology Model

5.2. Technology agnostic TE Topology model

The TE Topology model proposed in this document is meant to be network technology agnostic. Other technology specific TE Topology models can augment and use the building blocks provided by the proposed model.

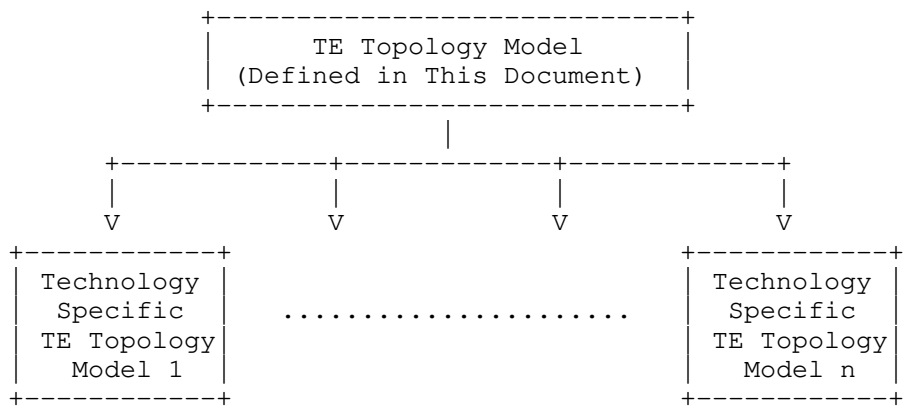


Figure 13: Augmenting the Technology agnostic TE Topology model

5.3. Model Structure

The high-level model structure proposed by this document is as shown below:

```
module: ietf-te-topology
augment /nw:networks/nw:network/nw:network-types:
   +--rw te-topology!

augment /nw:networks:
   +--rw te!
      +--rw templates
         +--rw node-template* [name] {template}?
         |  .............
         +--rw link-template* [name] {template}?
            .............

augment /nw:networks/nw:network:
   +--rw te-topology-identifier
   |  +--rw provider-id?   te-global-id
   |  +--rw client-id?     te-global-id
   |  +--rw topology-id?   te-topology-id
   +--rw te!
      |  .............

augment /nw:networks/nw:network/nw:node:
   +--rw te-node-id?   te-types:te-node-id
   +--rw te!
      |  .............
      +--rw tunnel-termination-point* [tunnel-tp-id]
```

```
            +--rw tunnel-tp-id    binary
            |  .............
            +--rw supporting-tunnel-termination-point* [node-ref tunnel-
   tp-ref]
               |  .............


   augment /nw:networks/nw:network/nt:link:
      +--rw te!
         |  ..........

   augment /nw:networks/nw:network/nw:node/nt:termination-point:
      +--rw te-tp-id?   te-types:te-tp-id
      +--rw te!
         |  ............
```

5.4. Topology Identifiers

The TE-Topology is uniquely identified by a key that has 3 constituents - topology-id, provider-id and client-id. The combination of provider-id and topology-id uniquely identifies a native TE Topology on a given provider. The client-id is used only when Customized TE Topologies come into play; a value of "0" is used as the client-id for native TE Topologies.

```
   augment /nw:networks/nw:network:
      +--rw te-topology-identifier
      |  +--rw provider-id?   te-global-id
      |  +--rw client-id?     te-global-id
      |  +--rw topology-id?   te-topology-id
      +--rw te!
         |  ............
```

5.5. Generic TE Link Attributes

The model covers the definitions for generic TE Link attributes - bandwidth, admin groups, SRLGs, switching capabilities, TE metric extensions etc.

```
     +--rw te-link-attributes
        ......................
        +--rw admin-status?                    te-admin-status
        |  ......................
        +--rw link-index?                      uint64
        +--rw administrative-group?            te-types:admin-groups
        +--rw link-protection-type?            enumeration
        +--rw max-link-bandwidth?              te-bandwidth
```

```
        +--rw max-resv-link-bandwidth?          te-bandwidth
        +--rw unreserved-bandwidth* [priority]
        |  ......................
        +--rw te-default-metric?                uint32
        |  ......................
        +--rw te-srlgs
        +--rw te-nsrlgs {nsrlg}?        ......................
```

5.6. Generic TE Node Attributes

The model covers the definitions for generic TE Node attributes.

The definition of a generic connectivity matrix is shown below:

```
     +--rw te-node-attributes
        ...........
        +--rw connectivity-matrices
        ...........
        |  +--rw connectivity-matrix* [id]
        |  |  +--rw id            uint32
        |  |  +--rw from
        |  |  |  +--rw tp-ref?        leafref
        |  |  |  +--rw label-restrictions
        |  |  +--rw to
        |  |  |  +--rw tp-ref?        leafref
        |  |  |  +--rw label-restrictions
        |  |  +--rw is-allowed?   boolean
        ...........
        |  |  +--rw underlay! {te-topology-hierarchy}?
        ...........
        |  |  +--rw path-constraints
        ...........
        |  |  +--rw optimizations
        ...........
        |  |  +--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 cooresponding 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 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.";
      }
      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:

- /nw:networks/nw:network/nw:network-types/tet:te-topology
  Unauthorized access to this subtree can disclose the TE topology type.

- /nw:networks/tet:te
  Unauthorized access to this subtree can disclose the TE node templates and TE link templates.

- /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.

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

- /nw:networks/nw:network/nt:link/tet:te
  Unauthorized access to this subtree can disclose the operational state information of TE links.

- /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
```

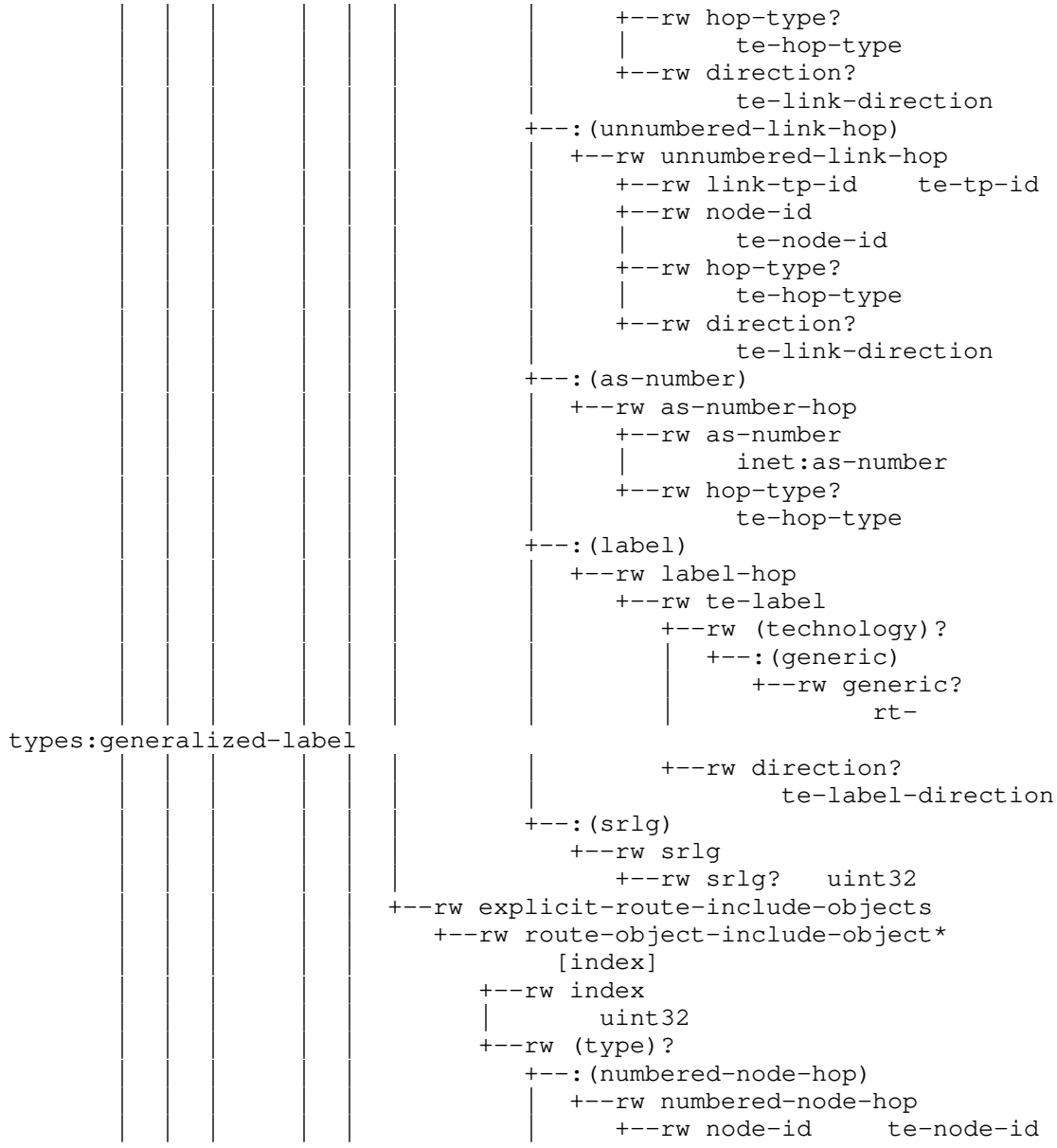
```
                  |     +--rw (technology)?
                  |        +--:(generic)
                  |           +--rw generic?   te-bandwidth
                  +--rw max-resv-link-bandwidth
                  |  +--rw te-bandwidth
                  |     +--rw (technology)?
                  |        +--:(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

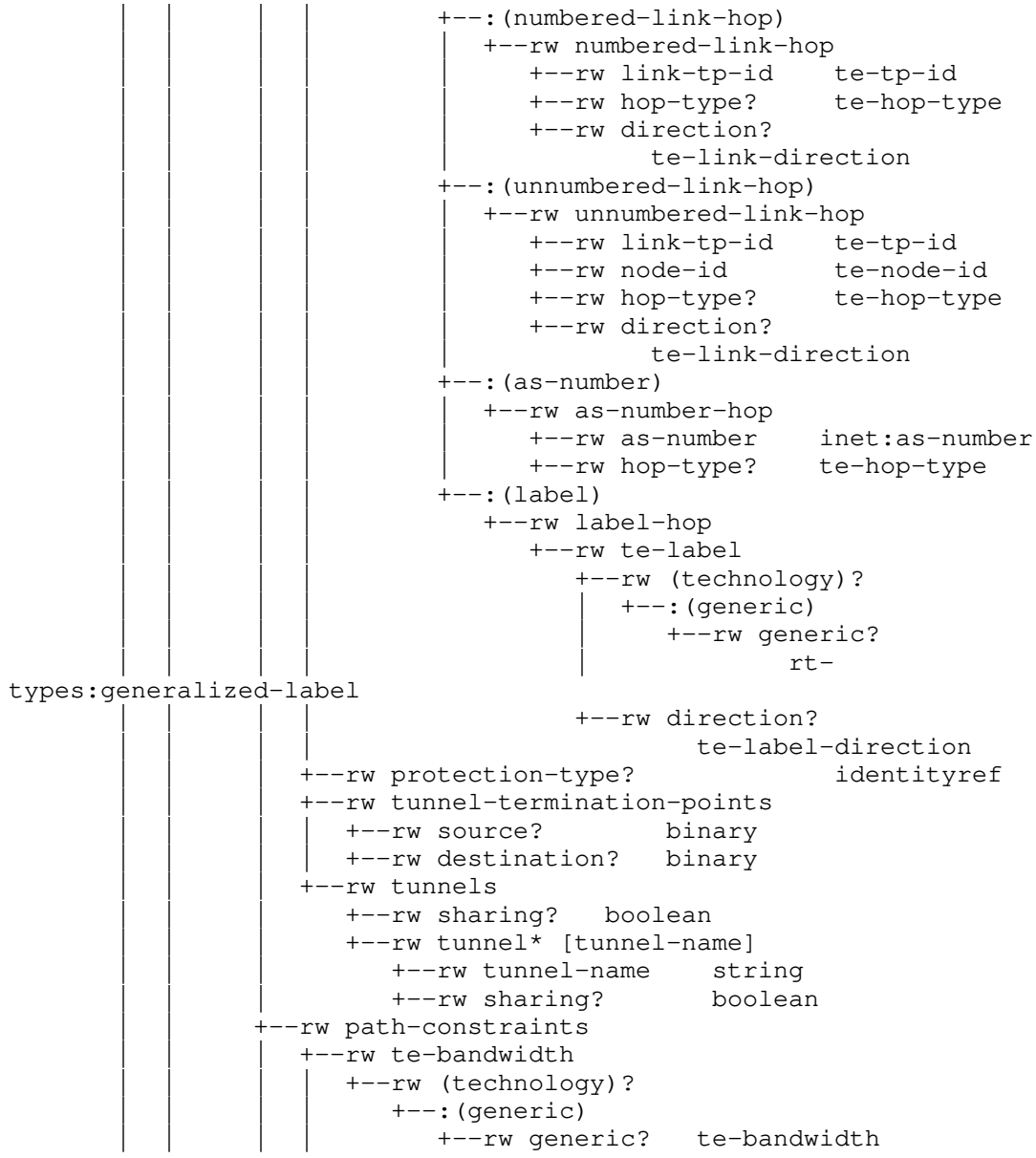
```
          |  |  |  |  |  |           |     +--rw hop-type?   te-hop-type
          |  |  |  |  |  |           +--:(numbered-link-hop)
          |  |  |  |  |  |           |  +--rw numbered-link-hop
          |  |  |  |  |  |           |     +--rw link-tp-id    te-tp-id
          |  |  |  |  |  |           |     +--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
```

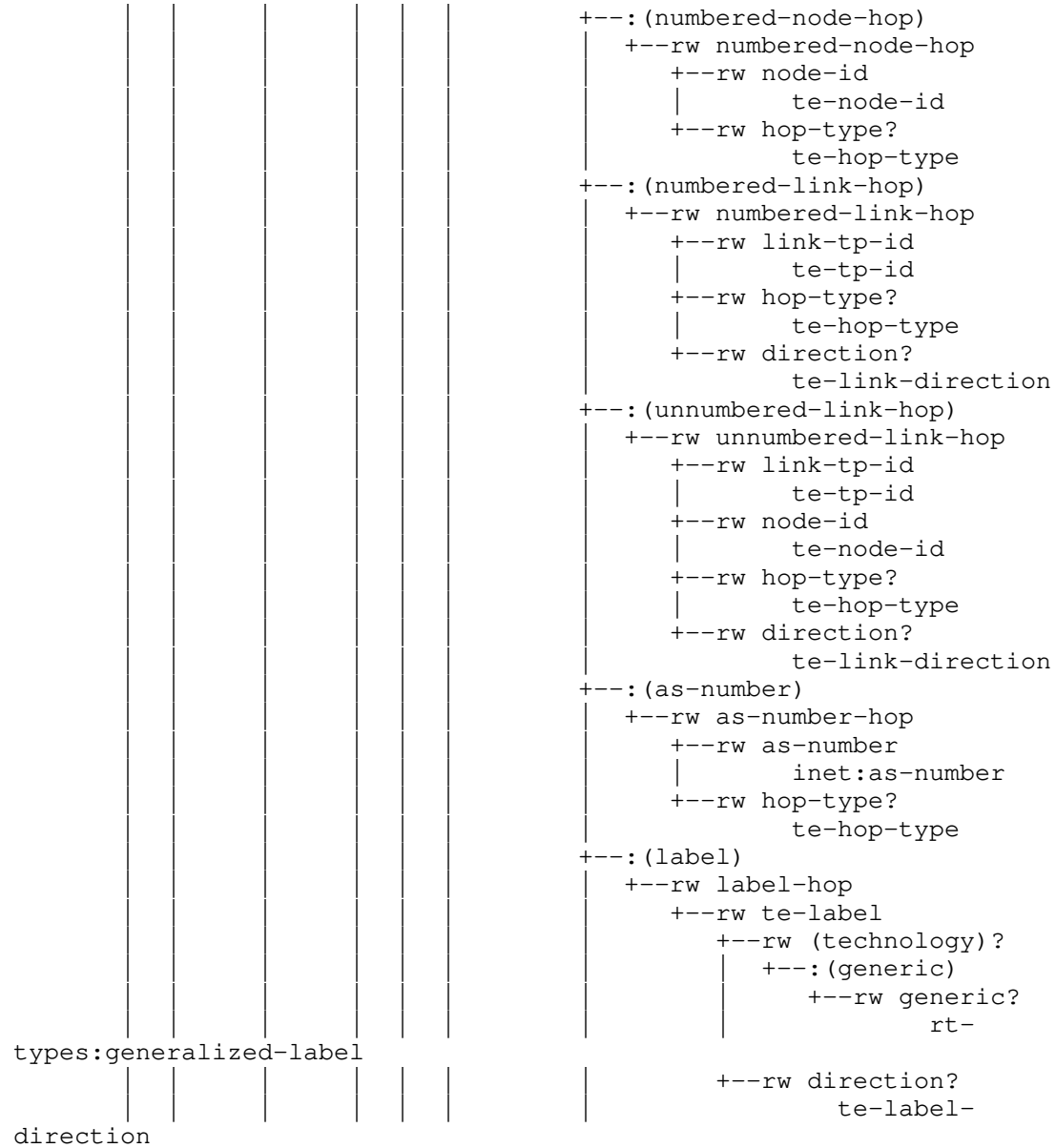
```
          |  |  |           |     +--ro as-number    inet:as-number
          |  |  |           |     +--ro hop-type?    te-hop-type
          |  |  |           +--:(label)
          |  |  |              +--ro label-hop
          |  |  |                 +--ro te-label
          |  |  |          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

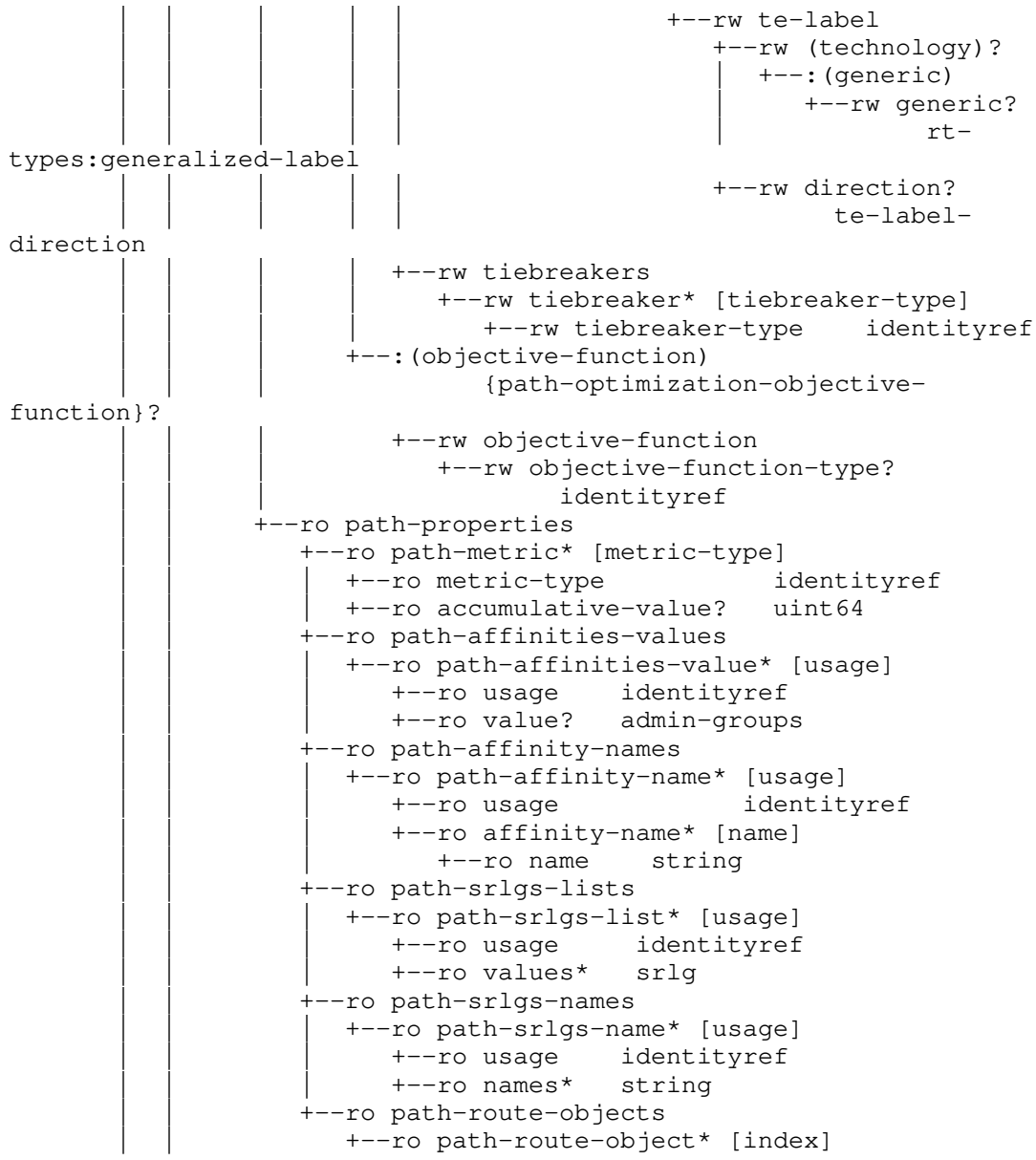
```
          |  |      |  |          |  +--rw numbered-node-hop
          |  |      |  |          |     +--rw node-id     te-node-id
          |  |      |  |          |     +--rw hop-type?   te-hop-type
          |  |      |  |          +--:(numbered-link-hop)
          |  |      |  |          |  +--rw 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
```

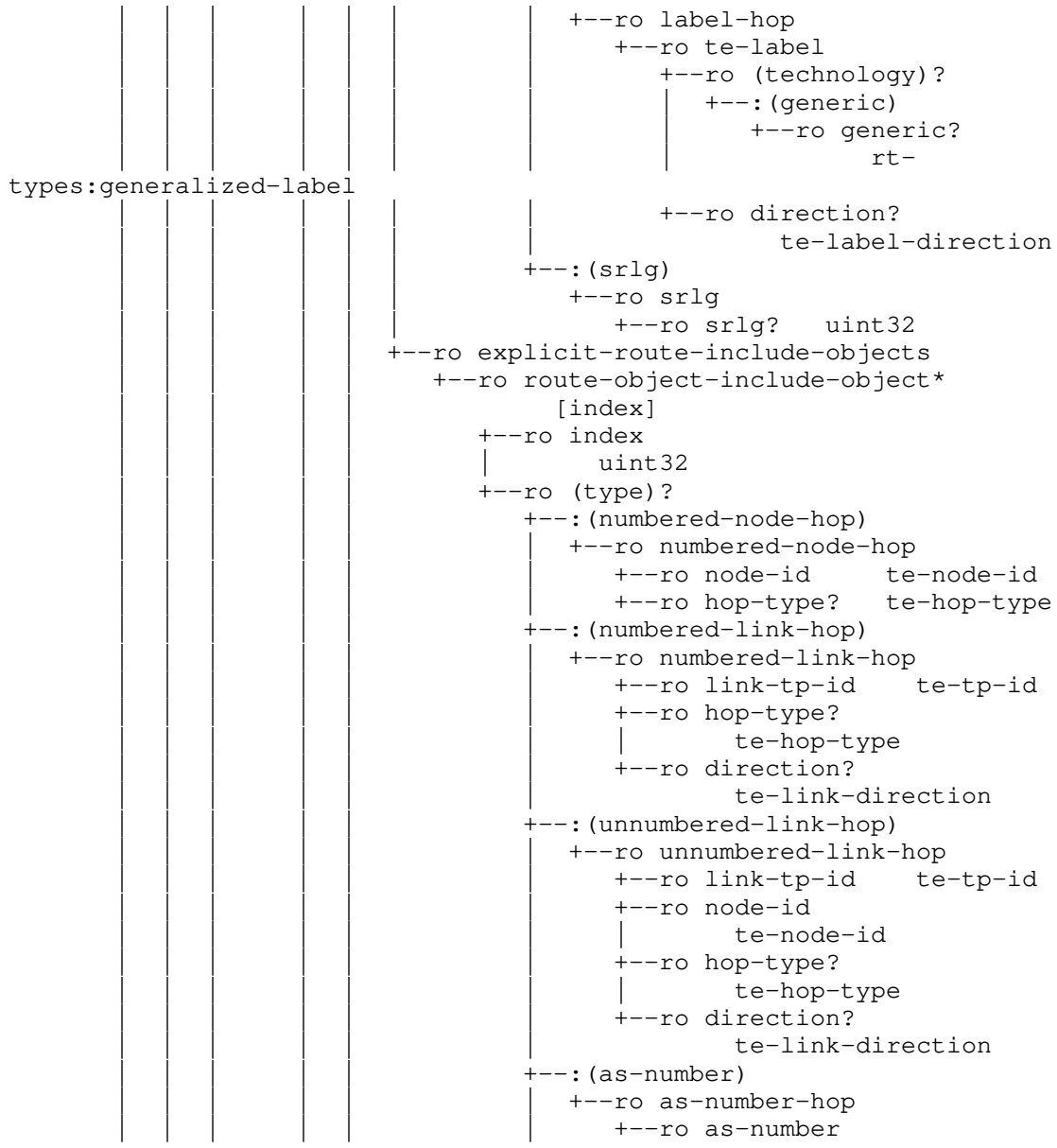
```
          |  +--ro altitude?    int64
          |  +--ro latitude?    geographic-coordinate-degree
          |  +--ro longitude?   geographic-coordinate-degree
          +--ro is-multi-access-dr?            empty
          +--ro information-source?            te-info-source
          +--ro information-source-instance?   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)?
```

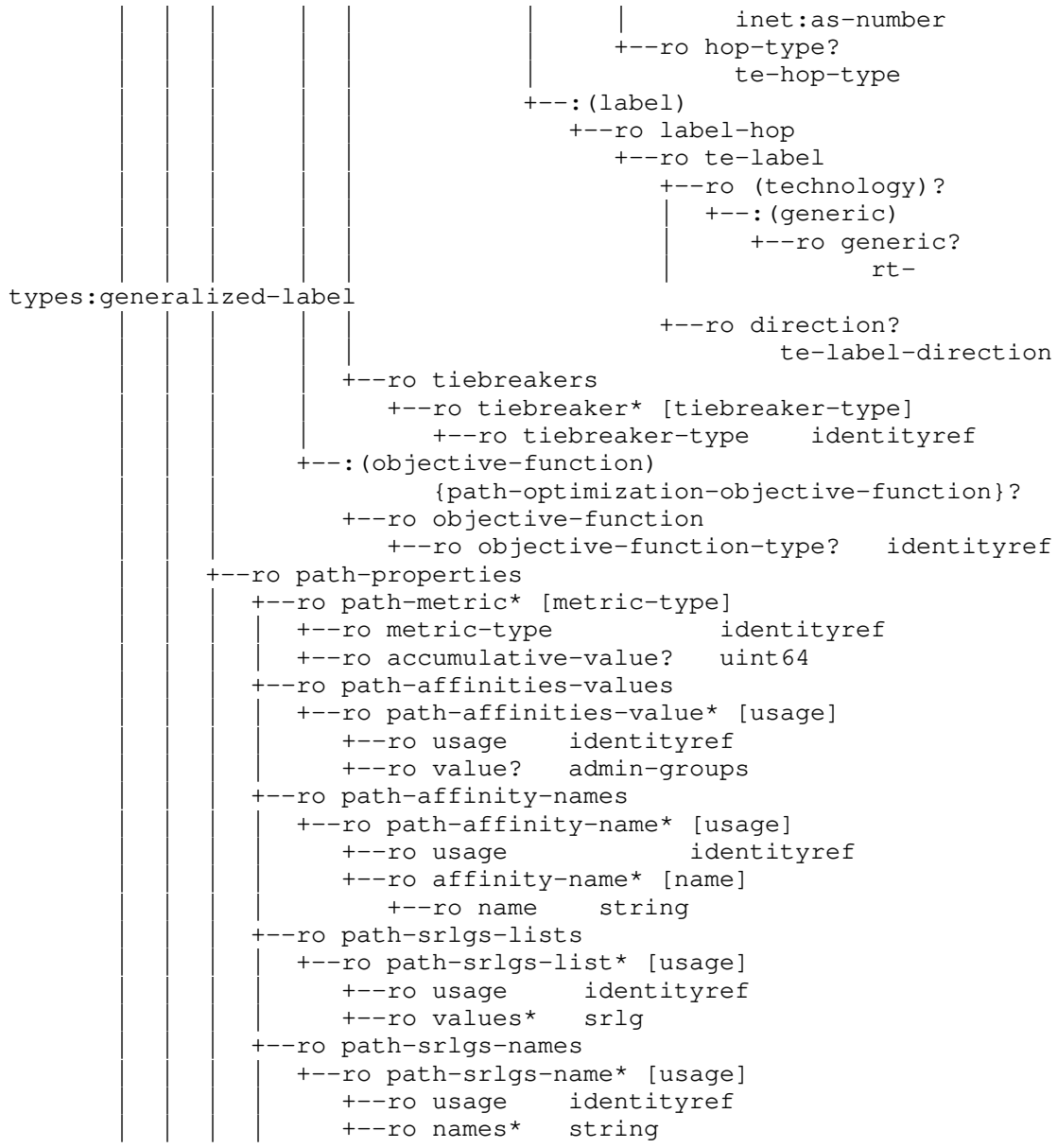
```
|  |  |  |  |  |  +--:(generic)
|  |  |  |  |  |     +--ro generic?
|  |  |  |  |  |              rt-types:generalized-label
|  |  |  |  |  +--ro direction?       te-label-direction
|  |  |  |  +--ro label-step
|  |  |  |  |  +--ro (technology)?
|  |  |  |  |     +--:(generic)
|  |  |  |  |        +--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
```

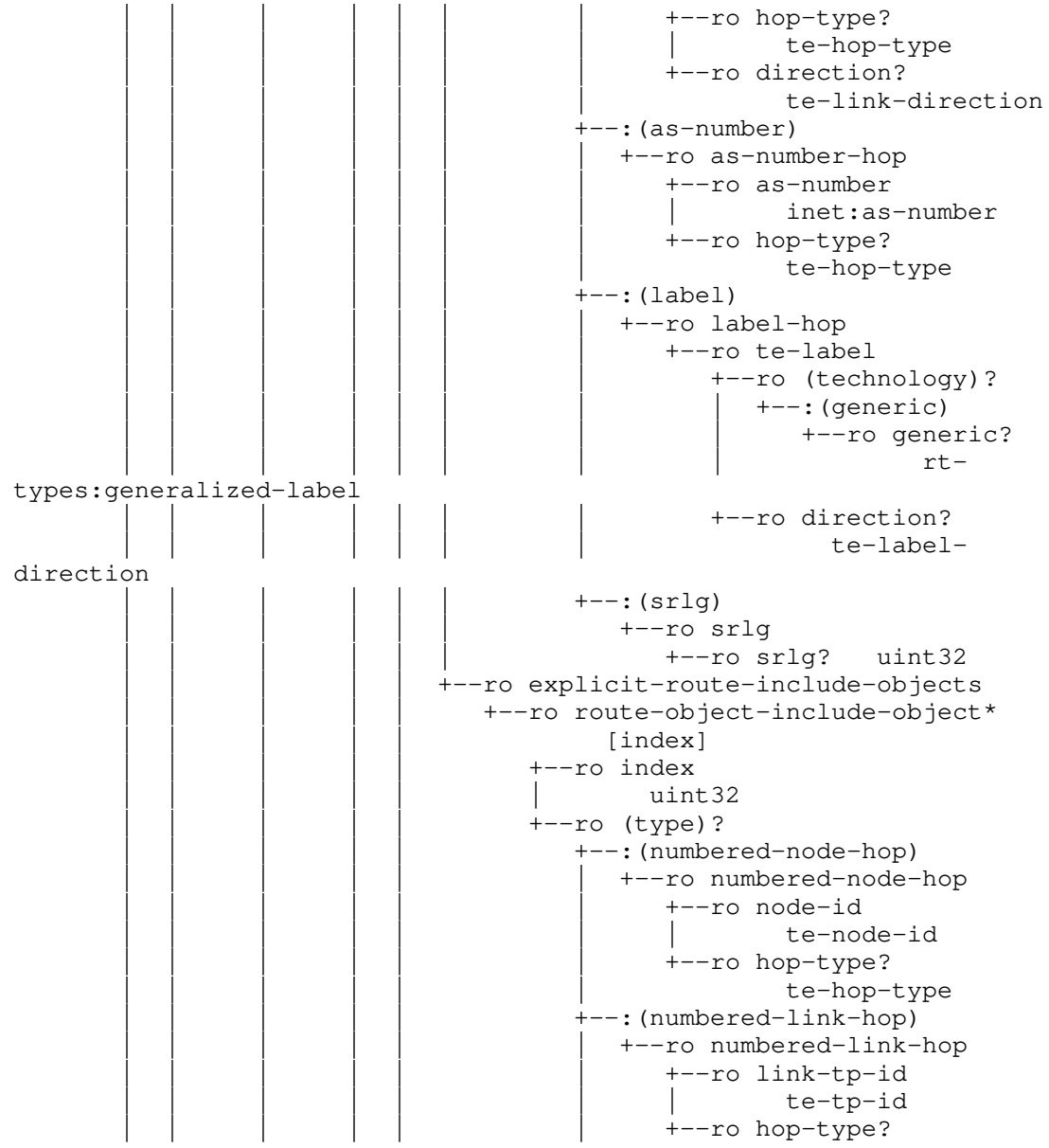
```
          |  |  |        +--ro label-start
          |  |  |        |  +--ro te-label
          |  |  |        |     +--ro (technology)?
          |  |  |        |     |  +--:(generic)
          |  |  |        |     |     +--ro generic?
          |  |  |        |     |              rt-types:generalized-
   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)?
```

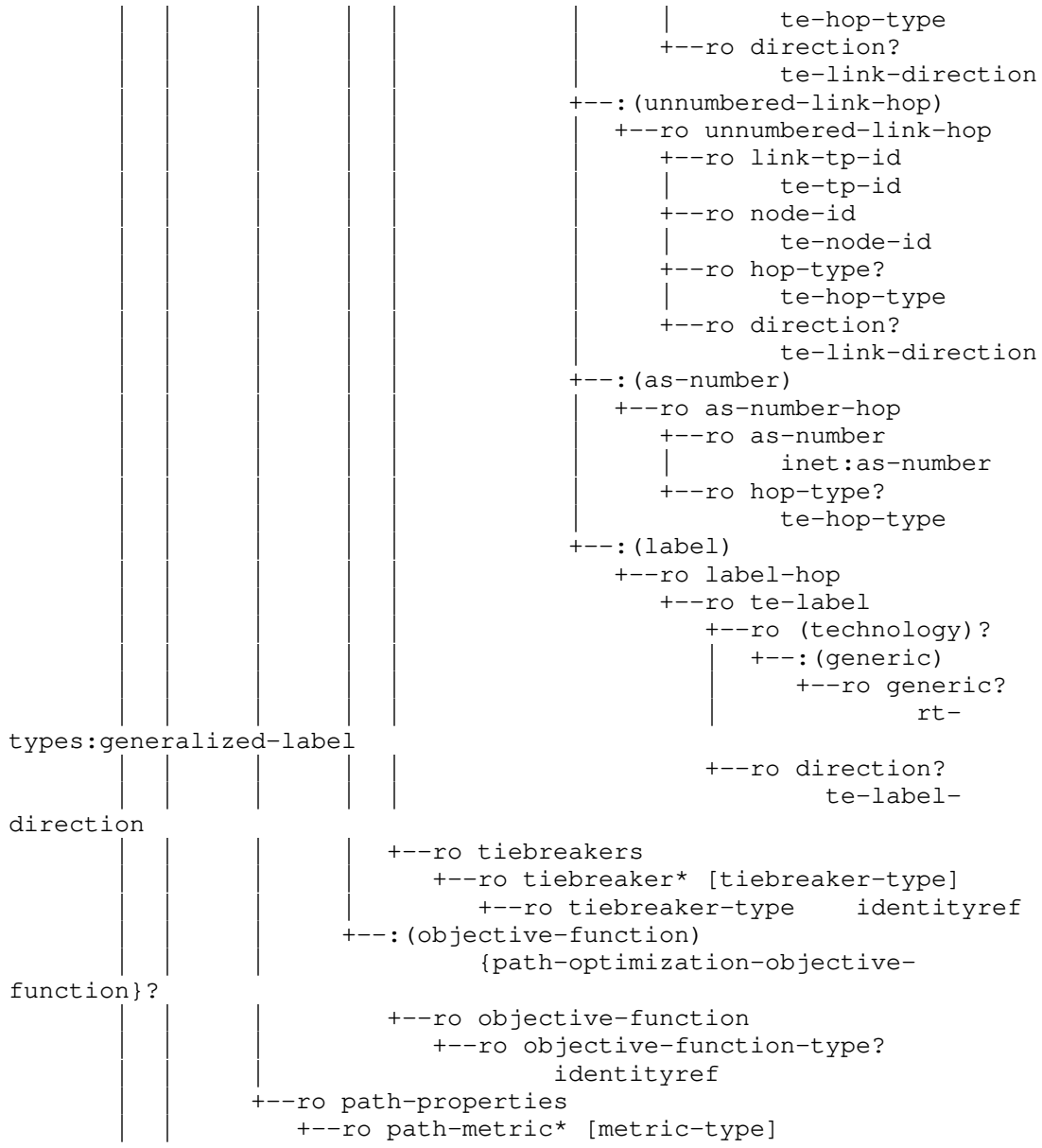
```
          |  |  |      |        |     |  +--:(generic)
          |  |  |      |        |     |     +--ro generic?
          |  |  |      |        |     |             rt-types:generalized-
   label
          |  |  |      |        |     +--ro direction?
          |  |  |      |        |             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)
```

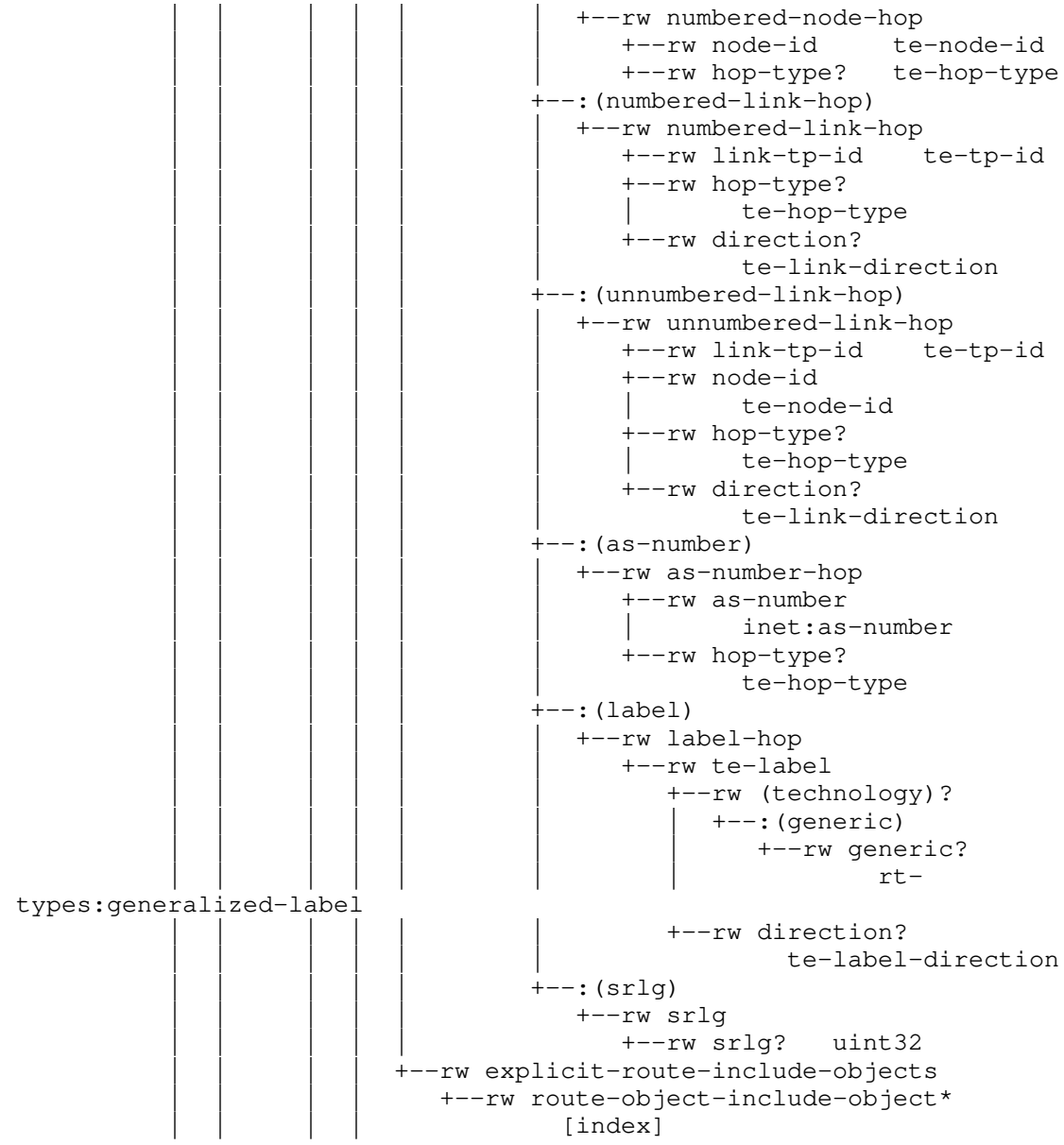
```
          |  |                  |  +--ro as-number-hop
          |  |                  |     +--ro as-number    inet:as-number
          |  |                  |     +--ro hop-type?    te-hop-type
          |  |                  +--:(label)
          |  |                     +--ro label-hop
     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

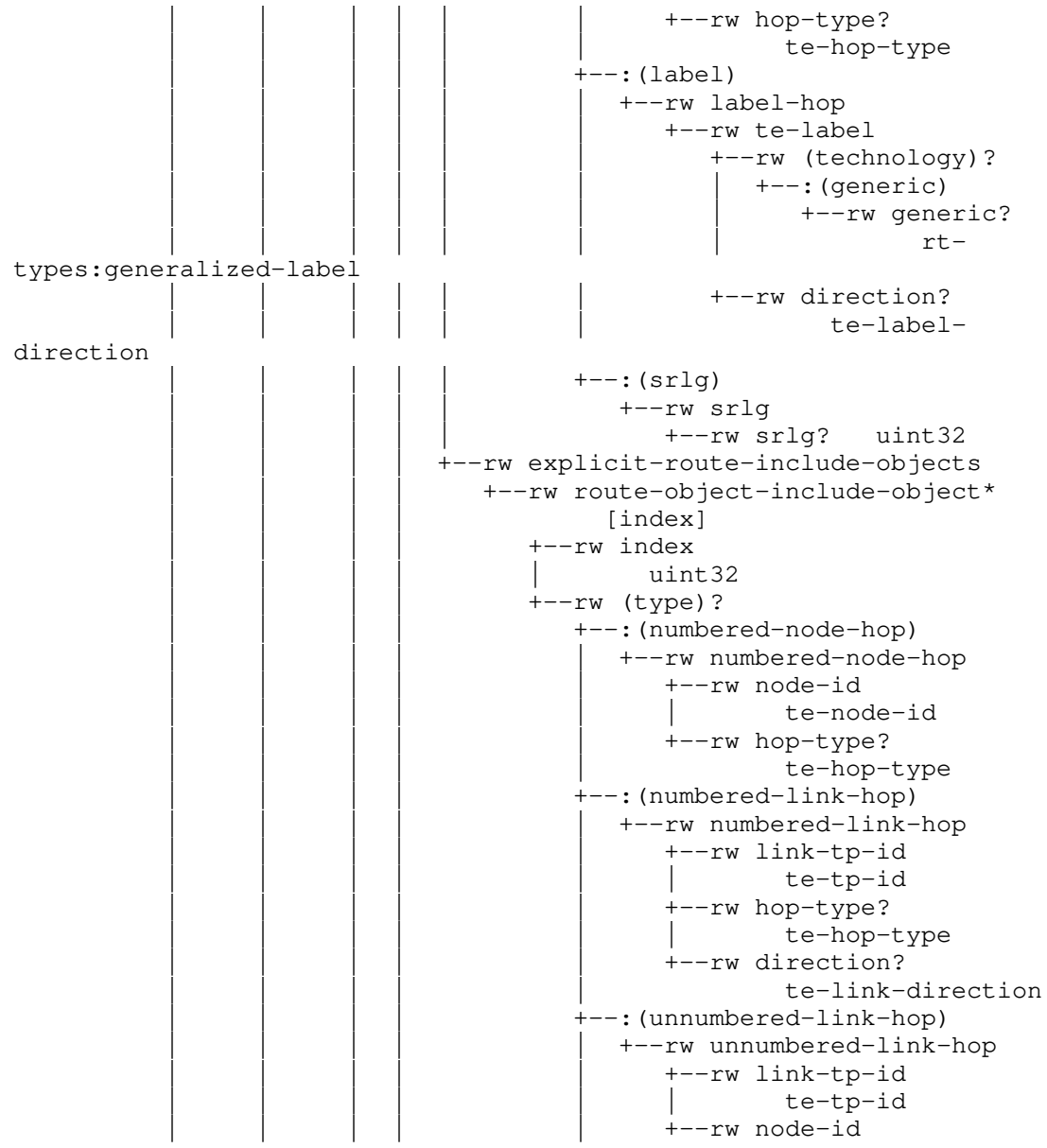
```
                |  |           |     +--ro link-tp-id    te-tp-id
                |  |           |     +--ro node-id       te-node-id
                |  |           |     +--ro hop-type?     te-hop-type
                |  |           |     +--ro direction?    te-link-direction
                |  |           +--:(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)?
```

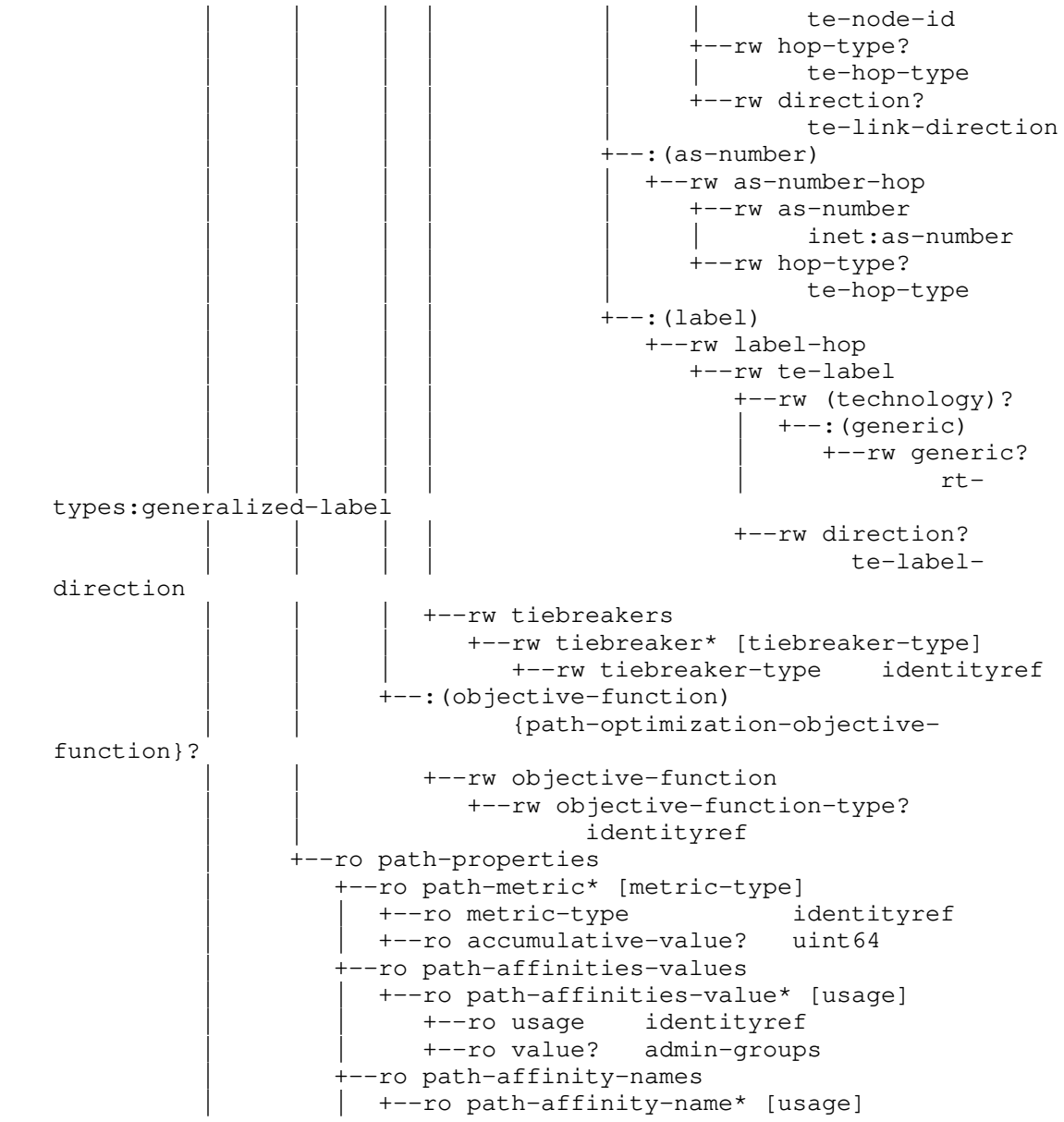

+--:(generic)
+--rw generic? int32
+--rw range-bitmap? yang:hex-string
+--rw is-allowed? boolean
+--rw underlay {te-topology-hierarchy}?
+--rw enabled? boolean
+--rw primary-path
+--rw network-ref?
-> /nw:networks/network/network-id
+--rw path-element* [path-element-id]
+--rw path-element-id uint32
+--rw (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

Teas Working Group
Internet Draft

Young Lee
Huawei

Intended status: Informational

Sergio Belotti
Nokia

Expires: April 2017

Dhruv Dhody
Huawei

Daniele Ceccarelli
Ericsson

Bin Young Yun
ETRI

October 24, 2016

Information Model for Abstraction and Control of TE Networks (ACTN)

draft-leebelotti-teas-actn-info-05.txt

Abstract

This draft provides an information model for Abstraction and Control of Traffic Engineered (TE) networks (ACTN).

Status of this Memo

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

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

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

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

Table of Contents

1. Introduction

This draft provides an information model for the requirements identified in the ACTN requirements [ACTN-Req] and the ACTN interfaces identified in the ACTN architecture and framework document [ACTN-Frame].

The purpose of this draft is to put all information elements of ACTN in one place before proceeding to development work necessary for protocol extensions and data models.

The ACTN reference architecture identified a three-tier control hierarchy as depicted in Figure 1:

- Customer Network Controllers (CNC)
- Multi-Domain Service Coordinator (MDSC)
- Physical Network Controllers (PNC).

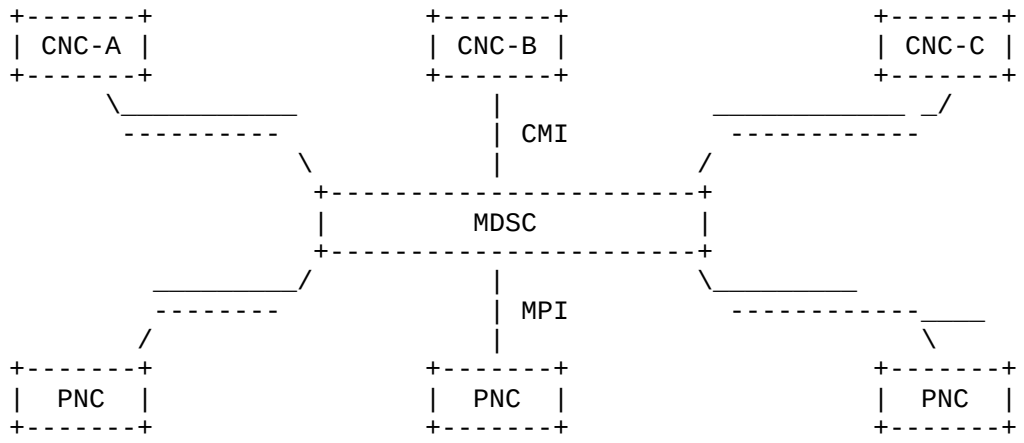


Figure 1: A Three-tier ACTN control hierarchy

The two interfaces with respect to the MDSC, one north of the MDSC and the other south of the MDSC are referred to as CMI (CNC-MDSC Interface) and MPI (MDSC-PNC Interface), respectively. It is intended to model these two interfaces and derivative interfaces thereof (e.g., MDSC to MSDC in a hierarchy of MDSCs) with one common model.

Appendix A provides some relevant ACTN use-cases extracted from [ACTN-Req]. Appendix A is information only and may help readers understand the context of key use-cases addressed in [ACTN-Req].

1.1. Terminology

- o A Virtual Network is a client view (typically a network slice) of the transport network. It is presented by the provider as a set of physical and/or abstracted resources. Depending on the agreement between client and provider various VN operations and VN views are possible. There are three aspects related to VN:

    1) VN Creation: VN could be pre-configured and created via static negotiation between customer and provider. In other cases, VN could also be created dynamically based

on the request from the customer with given SLA attributes which satisfy the customer's objectives.

2) Dynamic Operations: VN could be further modified and deleted based on customer request to request changes in the network resources reserved for the customer. The customer can further act upon the virtual network resources to perform E2E tunnel management (set-up/release/modify). These changes will incur subsequent LSP management on the operator's level.

3) VN View: (a) VN can be seen as an (or set of) e2e tunnel(s) from a customer point of view where an e2e tunnel is referred as a VN member. Each VN member (i.e., e2e tunnel) can then be formed by recursive aggregation of lower level paths at a provider level. Such end to end tunnels may comprise of customer end points, access links, intra domain paths and inter-domain link. In this view VN is thus a list of VN members. (b) VN can also be seen as a terms of topology comprising of physical and abstracted nodes and links. The nodes in this case include physical customer end points, border nodes, and internal nodes as well as abstracted nodes. Similarly the links includes physical access, inter-domain and intra-domain links as well as abstracted links. The abstracted nodes and links in this view can be pre-negotiated or created dynamically.

- o A Virtual Network Service (VNS) is the creation and offering of a Virtual Network by a provider to a customer in accordance with SLA agreements reached between them (e.g., re satisfying the customer's objectives).

- o Abstraction is the process of applying policy to the available TE information within a domain, to produce selective information that represents the potential ability to connect across the domain. Thus, abstraction does not necessarily offer all possible connectivity options, but it presents a general view of potential connectivity according to the policies that determine how the domain's administrator wants to allow the domain resources to be used [RFC7926].

- o Abstract topology: Every lower controller in the provider network, when is representing its network topology to a higher layer, it may want to selective hide details of the actual network topology, as suggested for abstraction in [RFC7926]. In such case, an abstract topology may be used for this purpose.

Abstract topology enhances scalability for the MDSC to operate multi-domain networks.

2. ACTN Common Interfaces Information Model

This section provides ACTN common interface information model to describe in terms of primitives, objects, their properties (represented as attributes), their relationships, and the resources for the service applications needed in the ACTN context.

Basic primitives (messages) are required between the CNC-MDSC and MDSC-PNC controllers. These primitives can then be used to support different ACTN network control functions like network topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options, etc.

The standard interface is described between a client controller and a server controller. A client-server relationship is recursive between a CNC and a MDSC and between a MDSC and a PNC. In the CMI, the client is a CNC while the server is a MDSC. In the MPI, the client is a MDSC and the server is a PNC. There may also be MDSC-MDSC interface(s) that need to be supported. This may arise in a hierarchy of MDSCs in which workloads may need to be partitioned to multiple MDSCs.

Basic primitives (messages) are required between the CNC-MDSC and MDSC-PNC controllers. These primitives can then be used to support different ACTN network control functions like network topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options, etc.

At a minimum, the following VN action primitives should be supported:

- VN Instantiate (See Section 2.1.1. for the description)

- VN Modify (See Section 2.1.2. for the description)

- VN Delete (See Section 2.1.3. for the description)

- VN Update ((See Section 2.1.4. for the description)

- VN Path Compute (See Section 2.1.5. for the description)

- VN Query (See Section 2.1.6. for the description)

In addition to VN action primitives, TE Update primitive should also be supported (See Section 2.1.7. for the description).

2.1. VN Action Primitives

This section provides a list of main primitives necessary to satisfy ACTN requirements specified in [ACTN-REQ].

<VN Action> describes main primitives. VN Action can be one of the following primitives: (i) VN Instantiate; (ii) VN Modify; (iii) VN Delete; (iv) VN Update; (v) VN Path Compute; (vi) VN Query.

```
<VN Action> ::= <VN Instantiate> |

                <VN Modify> |

                <VN Delete> |

                <VN Update> |

                <VN Path Compute> |

                <VN Query>
```

2.1.1. VN Instantiate

<VN Instantiate> refers to an action from customers/applications to request their VNs. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements. Please see the definition of VN in the section 2.

2.1.2. VN Modify

<VN Modify> refers to an action from customers/applications to modify an existing VN (i.e., instantiated VN). This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

2.1.3. VN Delete

<VN Delete> refers to an action from customers/applications to delete an existing VN. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

2.1.4. VN Update

<VN Update> refers to any update to the VN that need to be updated to the subscribers. VN Update fulfills a push model at CMI level, to make aware customers of any specific changes in the topology details related to VN instantiated.

Note the VN Update means the connection-related information (e.g., LSPs) update that has association with VNs.

2.1.5. VN Path Compute

<VN Path Compute> consists of Request and Reply. Request refers to an action from customers/applications to request a VN path computation. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

<VN Path Compute> Reply refers to the reply in response to <VN Path Compute> Request.

<VN Path Compute> Request/Reply is to be differentiated from a VN Instantiate. The purpose of VN Path Compute is a priori exploration to estimate network resources availability and getting a list of possible paths matching customer/applications constraints. To make this type of request Customer/application controller can have a shared (with lower controller) view of an abstract network topology on which to get the constraints used as input in a Path Computation request. The list of paths obtained by the request can be used by customer/applications to give path constrains during VNS connectivity request and to compel the lower level controller (e.g. MDSC) to select the path that Client/application controller has chosen among the set of paths returned by the Path Computation primitives. The importance of this primitives is for example in a scenario like multi-domain in which the optimal path obtained by an orchestrator as sum of optimal paths for different domain controller

cannot be the optimal path in the Client/application controller prospective. This only applies between CNC and MDSC.

2.1.6. VN Query

<VN Query> refers to any query pertaining to the VN that has been already instantiated. VN Query fulfills a pull model and permit to get topology view.

<VN Query Reply> refers to the reply in response to <VN Query>.

2.1.7. TE Update (for TE resources)

<TE Update> it is a primitives specifically related to MPI interface to provide TE resource update between any domain controller towards MDSC regarding the entire content of any "domain controller" TE topology or an abstracted filtered view of TE topology depending on negotiated policy.

<TE Update> ::= [<Abstraction>]<TE-topology...>

<TE-topology> ::= <TE-Topology-reference> <Node-list> <Link-list>

<Node-list> ::= <Node>[<Node-list>]

<Node> ::= <Node> <TE-Termination Points>

<Link-list> ::= <Link>[<Link-list>]

Where

<Abstraction> provides information on level of abstraction (as determined a priori).

<TE-topology-reference> ::= information related to the specific te-topology related to nodes and links present in this TE-topology.

<Node-list> ::= detailed information related to a specific node belonging to a te-topology e.g. te-node-attributes [TE-TOPO].

<Link-list> ::= information related to the specific link related belonging to a te-topology e.g. te-link-attributes [TE-TOPO].

<TE-Termination Points> ::= information details associated to the termination point of te-link related to a specific node e.g. interface-switching-capability [TE-TOPO].

2.2. VN Objects

This section provides a list of objects associated to VN action primitives.

2.2.1. VN Identifier

<VN Identifier> is a unique identifier of the VN.

2.2.2. VN Service Characteristics

VN Service Characteristics describes the customer/application requirements against the VNs to be instantiated.

<VN Service Characteristics> ::= <VN Connectivity Type>

(<VN Traffic Matrix>...)

<VN Survivability>

Where

<VN Connectivity Type> ::= <P2P>|<P2MP>|<MP2MP>|<MP2P>|<Multi-destination>

The Connectivity Type identifies the type of required VN Service. In addition to the classical type of services (e.g. P2P/P2MP etc.), ACTN defines the "multi-destination" service that is a new P2P service where the end points are not fixed. They can be chosen among a list of pre-configured end points or dynamically provided by the CNC.

<VN Traffic Matrix> ::= <Bandwidth>

[<VN Constraints>]

The VN Traffic Matrix represents the traffic matrix parameters required against the service connectivity required and so the VN request instantiation between service related Access Points [ACTN-Frame]. Bandwidth is a mandatory parameter and a number of optional constrains can be specified in the <VN Constrains> (e.g. diversity,

cost). They can include objective functions and TE metrics bounds as specified in [RFC5441].

Further details on the VN constraints are specified below:

```
<VN Constraints> ::= [<Layer Protocol>]

                     [<Diversity>]

                     [<Shared Risk>]

                     <Metric>
```

Where:

<Layer Protocol> Identifies the layer at which the VN service is requested. It could be for example MPLS, ODU, and OCh.

<Diversity> This allows asking for diversity constraints for a VN Instantiate/Modify or a VN Path Compute. For example, a new VN or a path is requested in total diversity from an existing one (e.g. diversity exclusion).

```
<Diversity> ::= <VN-exclusion> (<VN-id>...) |

                <VN-E2E Tunnel-exclusion> (<Tunnel-id>...)
```

<Shared Risk> Based on the realization of VN required, group of physical resources can be impacted by the same risk. An E2E tunnel can be impacted by this shared risk. This is used to get the SRLG associated with the different tunnels composing a VN.

<Metric> can include all the Metrics (cost, delay, delay variation, latency), bandwidth utilization parameters defined and referenced by [RFC3630] and [RFC7471].

<VN Survivability> describes all attributes related to the VN recovery level and its survivability policy enforced by the customers/applications.

```
<VN Survivability> ::= <VN Recovery Level>

                       [<VN Tunnel Recovery Level>]

                       [<VN Survivability Policy>]
```

Where:

<VN Recovery Level> It is a value representing the requested level of resiliency required against the VN. The following values are defined:

. Unprotected VN
. VN with per tunnel recovery: The recovery level is defined against the tunnels composing the VN and it is specified in the <VN Tunnel Recovery Level>.

<VN Tunnel Recovery Level> ::= <0:1>|<1+1>|<1:1>|<1:N>|<M:N>|

<On the fly restoration>

The VN Tunnel Recovery Level indicates the type of protection or restoration mechanism applied to the VN. It augments the recovery types defined in [RFC4427].

<VN Survivability Policy> ::= [<Local Reroute Allowed>]

[<Domain Preference>]

[<Push Allowed>]

[<Incremental Update>]

Where:

<Local Reroute Allowed> is a delegation policy to the Server to allow or not a local reroute fix upon a failure of the primary LSP.

<Domain Preference> is only applied on the MPI where the MDSC (client) provides a domain preference to each PNC (server).e.g. when a inter-domain link fails, then PNC can choose the alternative peering with this info.

<Push Allowed> is a policy that allows a server to trigger an updated VN topology upon failure without an explicit request from the client. Push action can be set as default unless otherwise specified.

<Incremental Update> is another policy that triggers an incremental update from the server since the last period of

update. Incremental update can be set as default unless otherwise specified.

2.2.3. VN End-Point

<VN End-Point> Object describes the VN's customer end-point characteristics.

<VN End-Point> ::= (<Access Point Identifier>

[<Access Link Capability>]

[<Source Indicator>])...

Where:

<Access point identifier> It represents a unique identifier of the client end-point. They are used by the customer to ask for the setup of a virtual network creation. A <VN End-Point> is defined against each AP in the network and is shared between customer and provider. Both the customer and the provider will map it against his own physical resources.

<Access Link Capability> An optional object that identifies the capabilities of the access link related to the given access point. (e.g., max-bandwidth, bandwidth availability, etc.)

<Source Indicator> indicates if an End-point is source or not.

2.2.4. VN Objective Function

The VN Objective Function applies to each VN member (i.e., each E2E tunnel) of a VN.

The VN Objective Function can reuse objective functions defined in [RFC5541] section 4.

For a single path computation, the following objective functions are defined:

- MCP is the Minimum Cost Path with respect to a specific metric (e.g. shortest path).

- o MLP is the Minimum Load Path, that means find a path composted by te-link least loaded.
- o MBP is the Maximum residual Bandwidth Path.

For a concurrent path computation, the following objective functions are defined:

- o MBC is to Minimize aggregate Bandwidth Consumption.
- o MLL is to Minimize the Load of the most loaded Link.
- o MCC is to Minimize the Cumulative Cost of a set of paths.

2.2.5. VN Action Status

<VN Action Status> is the status indicator whether the VN has been successfully instantiated, modified, or deleted in the server network or not in response to a particular VN action.

Note that this action status object can be implicitly indicated and thus not included in any of the VN primitives discussed in Section 2.3.

2.2.6. VN Associated LSP

<VN Associated LSP> describes the instantiated LSPs that is associated with the VN. <VN Associated LSP> is used between each domain PNC and the MDSC as part of VN Update once the VN is instantiated in each domain network and when CNC want to have more details about the topology instantiated as consequence of a VN Instantiate.

<VN Associated LSP> ::= <VN Identifier> (<LSP>...)

2.2.7. VN Computed Path

The VN Computed Path is the list of paths obtained after the VN path computation request from higher controller. Note that the computed path is to be distinguished from the LSP. When the computed path is signaled in the network (and thus the resource is reserved for that path), it becomes an LSP.

<VN Computed Path> ::= (<Path>...)

2.2.8. VN Service Preference

This section provides VN Service preference. VN Service is defined in Section 2.

<VN Service Preference> ::= [<Location Service Preference >]

[<Client-specific Preference >]

[<End-Point Dynamic Selection Preference >]

Where

<Location Service Preference describes the End-Point Location's (e.g. Data Centers) support for certain Virtual Network Functions (VNFs) (e.g., security function, firewall capability, etc.)and is used to find the path that satisfies the VNF constraint.

<Client-specific Preference> describes any preference related to Virtual Network Service (VNS) that application/client can enforce via CNC towards lower level controllers. For example, permission the correct selection from the network of the destination related to the indicated VNF It is e.g. the case of VM migration among data center and CNC can enforce specific policy that can permit MDSC/PNC to calculate the correct path for the connectivity supporting the data center interconnection required by application.

<End-Point Dynamic Selection Preference> describes if the End-Point (e.g. Data Center) can support load balancing, disaster recovery or VM migration and so can be part of the selection by MDSC following service Preference enforcement by CNC.

2.3. Mapping of VN Primitives with VN Objects

This section describes the mapping of VN Primitives with VN Objects based on Section 2.2.

<VN Instantiate> ::= <VN Service Characteristics>

<VN Objective Function>

<VN End-Point>

[<VN Service Preference>]

<VN Modify> ::= <VN identifier>
<VN Service Characteristics>
[<VN Objective Function>]
<VN End-Point>
[<VN Service Preference>]

<VN Delete> ::= <VN Identifier>

<VN Update> :: = <VN Identifier>
<VN Associated LSP>

<VN Path Compute Request> ::= <VN Service Characteristic>
<VN Objective Function>
<VN End-Point>

<VN Path Compute Reply> ::= <VN Computed Path>

<VN Query> ::= <VN Identifier>

<VN Query Reply> ::= <VN Identifier>
<VN Associated LSP>

3. References

3.1. Normative References

[DRAFT-SER-AWARE] Dhruv Dhody, Qin Wu, Vishwas Manral, Zafar Ali, and Kenji Kumaki, "Extensions to the Path Computation Element Communication Protocol (PCEP) to compute service aware Label Switched Path (LSP).," June 2016, draft-ietf-pce-pcep-service-aware-10.

3.2. Informative References

[TE-TOPO] Liu, X. et al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress.Informative References

[ACTN-Req] Y. Lee, et al., "Requirements for Abstraction and Control of Transport Networks", draft-lee-teas-actn-requirements, work in progress.

[ACTN-Frame] D. Ceccarelli, et al., "Framework for Abstraction and Control of Transport Networks", draft-ceccarelli-teas-actn-framework, work in progress.

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

[RFC5541] JL. Le Roux, JP. Vasseur and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, June 2009.

[RFC7926] A.Farrel, et al., "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.

4. Contributors

Contributors' Addresses

Authors' Addresses

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

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

Dhruv Dhody
Huawei Technologies,
Divyashree Technopark, Whitefield
Bangalore, India
Email: dhruv.ietf@gmail.com

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

Bin Young Yun
ETRI
Email: byyun@etri.re.kr

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

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

Appendix A: ACTN Applications

A.1. Coordination of Multi-destination Service Requirement/Policy

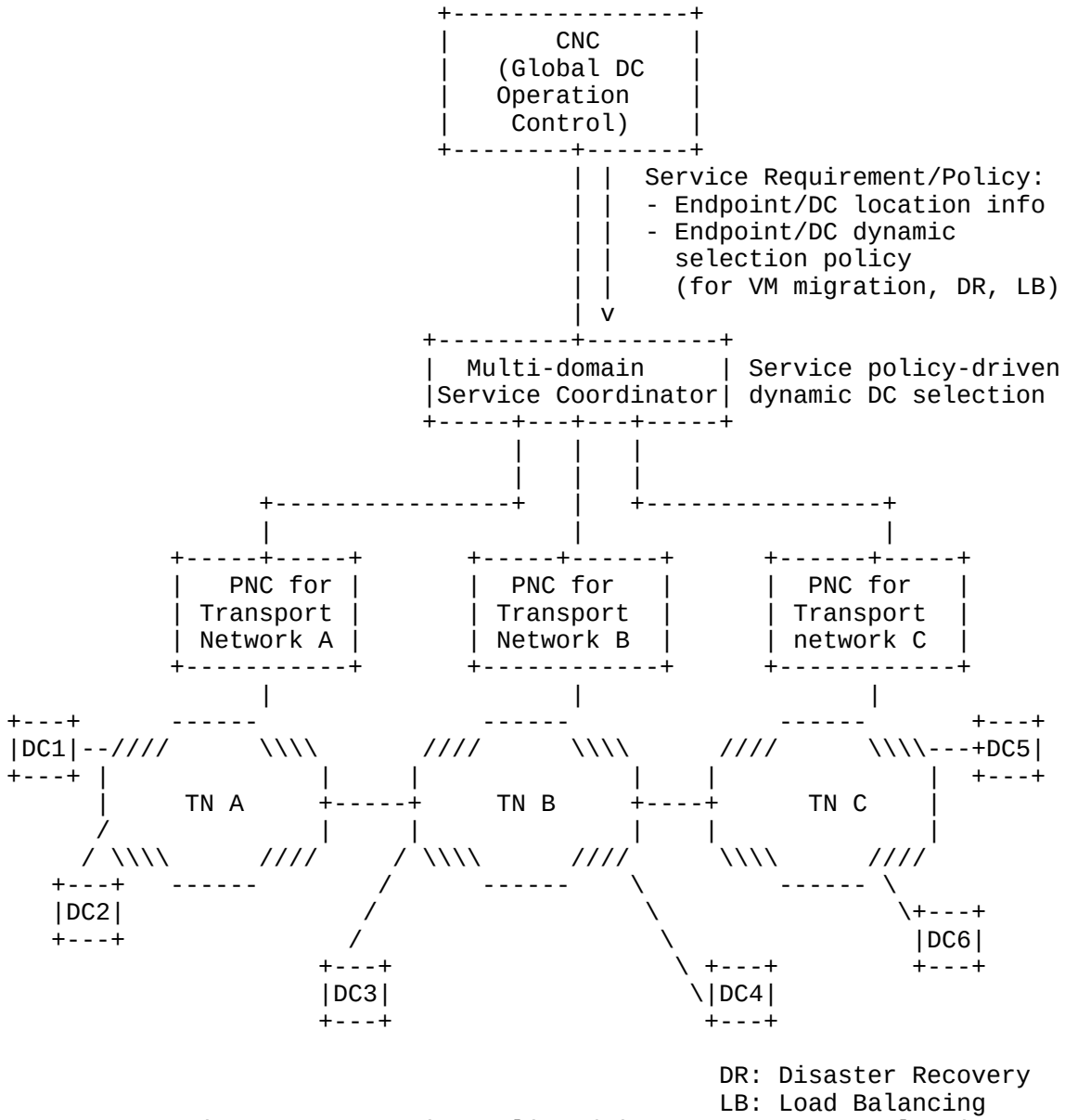


Figure A.1: Service Policy-driven Data Center Selection

Figure A.1 shows how VN service policies from the CNC are incorporated by the MDSC to support multi-destination applications. Multi-destination applications refer to applications in which the selection of the destination of a network path for a given source needs to be decided dynamically to support such applications.

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

A.2. Application Service Policy-aware Network Operation

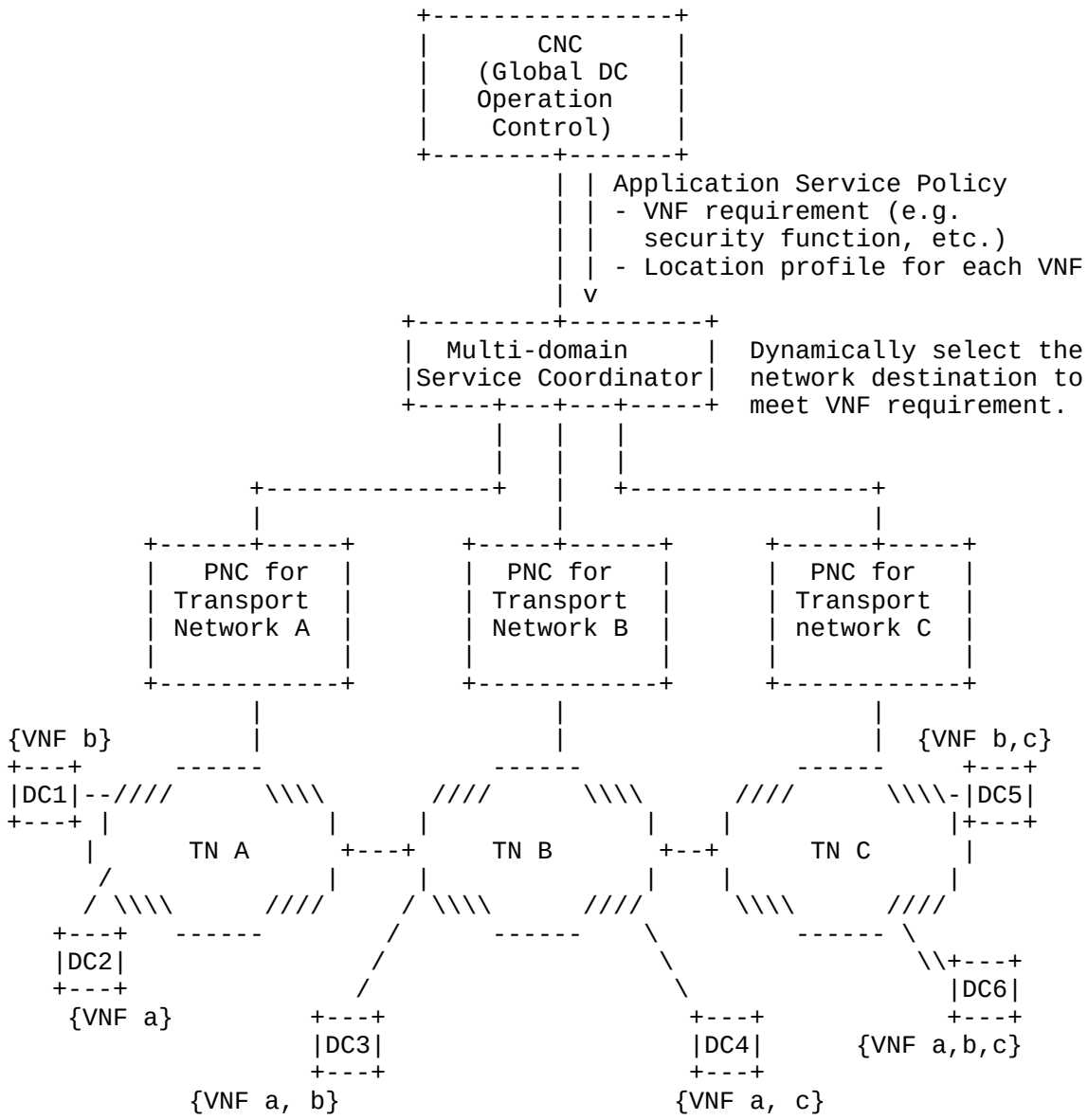


Figure A.2: Application Service Policy-aware Network Operation

This scenario is similar to the previous case in that the VN service policy for the application can be met by a set of multiple destinations that provide the required virtual network functions (VNF). Virtual network functions can be, for example, security functions required by the VN application. The VN service policy by the CNC would indicate the locations of a certain VNF that can be fulfilled. This policy information is critical in finding the optimal network path subject to this constraint. As VNFs can be dynamically moved across different DCs, this policy should be dynamically enforced from the CNC to the MDSC and the PNCs.

A.3. Network Function Virtualization Service Enabled Connectivity

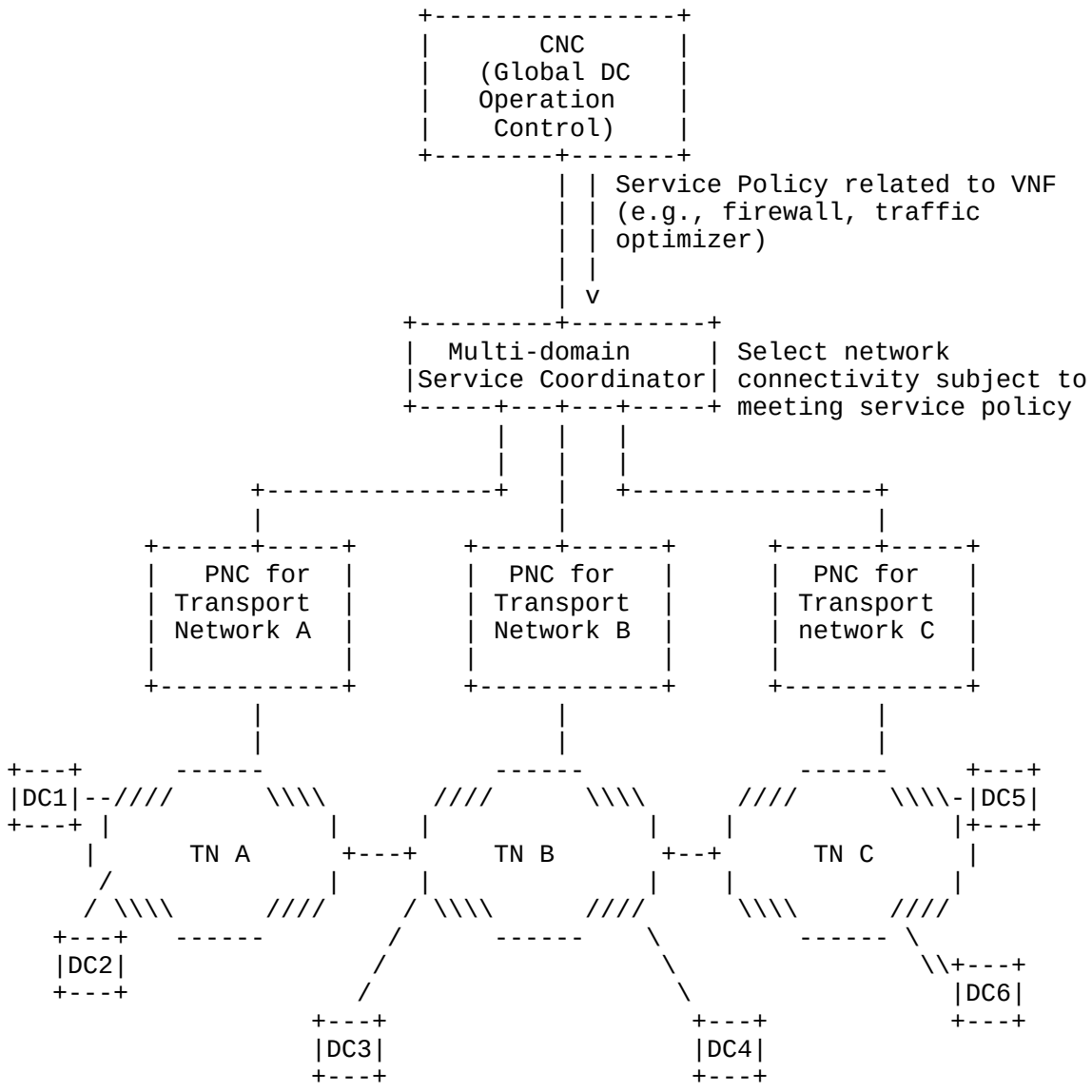


Figure A.3: Network Function Virtualization Service Enabled Connectivity

Network Function Virtualization 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. The customer has to provide (and CNC is providing this)the type of VNF he needs and the policy associated with it (e.g. metric like estimated delay to reach where VNF is located in the DC). The policy linked to VNF is requested as part of the VN instantiation. 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).

A.4. Dynamic Service Control Policy Enforcement for Performance and Fault Management

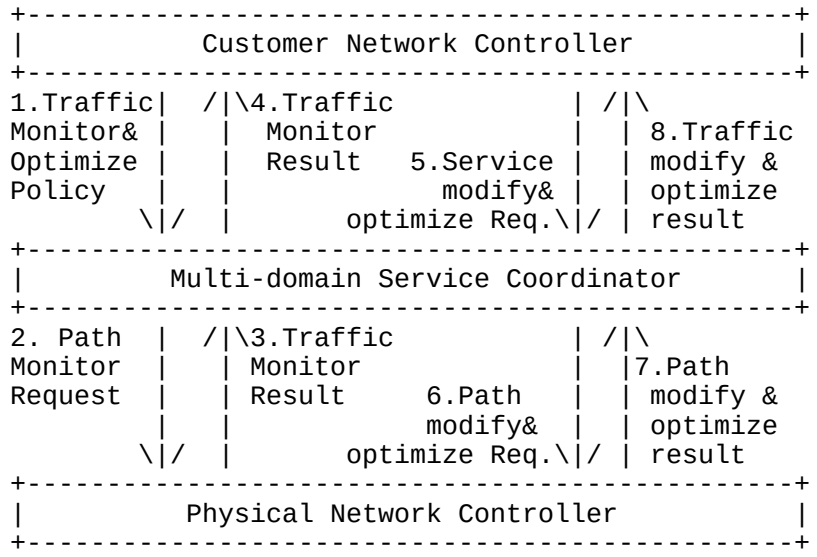


Figure A.4: Dynamic Service Control for Performance and Fault Management

Figure A.4 shows the flow of dynamic service control policy enforcement for performance and fault management initiated by customer per VN. The feedback loop and filtering mechanism tailored for VNs performed by the MDSC differentiates this ACTN scope from traditional network management paradigm. VN level dynamic OAM data model is a building block to support this capability.

A.5. E2E VN Survivability and Multi-Layer (Packet-Optical) Coordination for Protection/Restoration

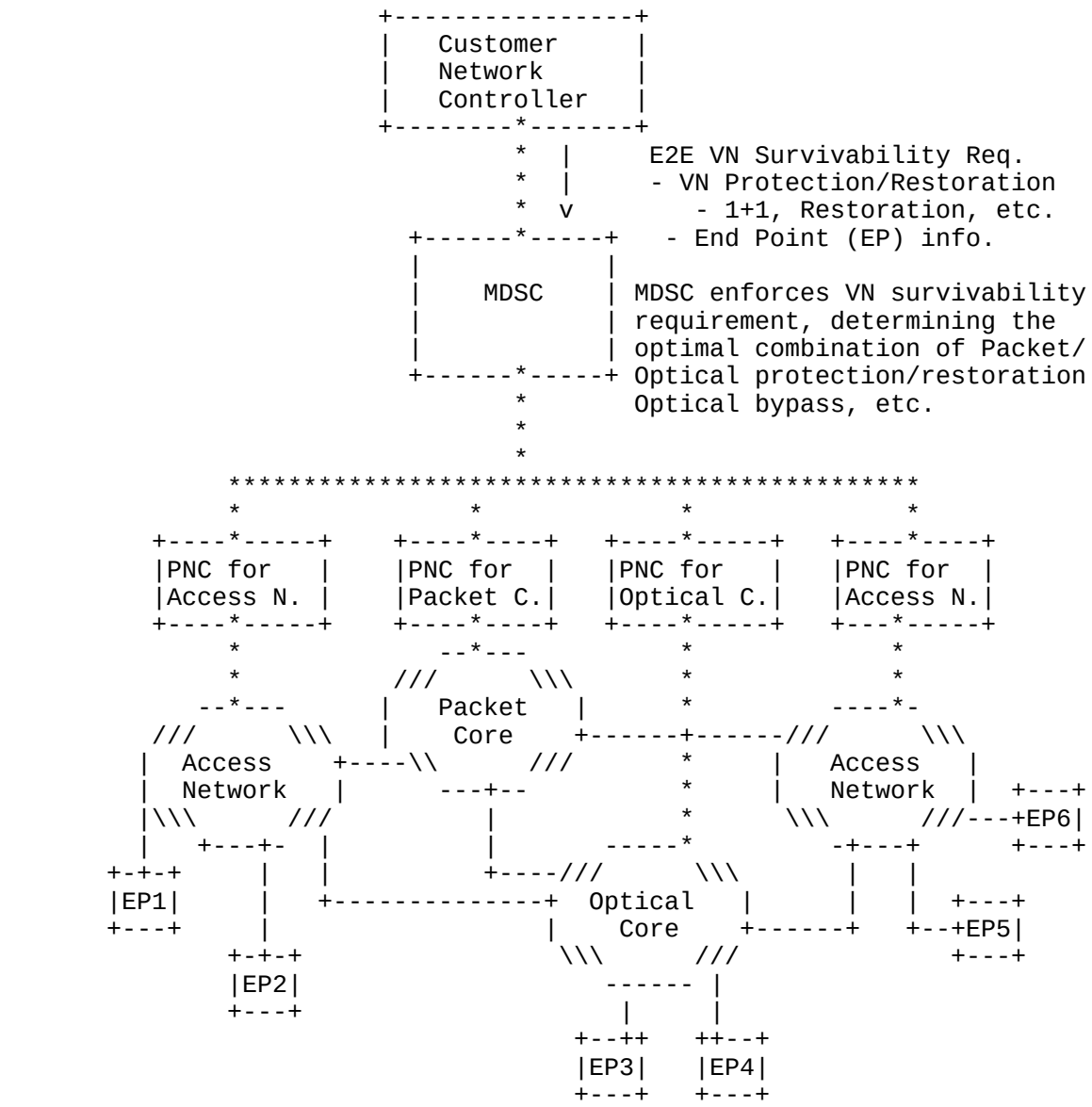


Figure A.5: E2E VN Survivability and Multi-layer Coordination for Protection and Restoration

Figure A.5 shows the need for E2E protection/restoration control coordination that involves CNC, MDSC and PNCs to meet the VN survivability requirement. VN survivability requirement and its policy need to be translated into multi-domain and multi-layer network protection and restoration scenarios across different controller types. After an E2E path is setup successfully, the MDSC has a unique role to enforce policy-based flexible VN survivability requirement by coordinating all PNC domains.

As seen in Figure A.5, multi-layer (i.e., packet/optical) coordination is a subset of this E2E protection/restoration control operation. The MDSC has a role to play in determining an optimal protection/restoration level based on the customer's VN survivability requirement. For instance, the MDSC needs to interface the PNC for packet core as well as the PNC for optical core and enforce protection/restoration policy as part of the E2E protection/restoration. Neither the PNC for packet core nor the PNC for optical core is in a position to be aware of the E2E path and its protection/restoration situation. This role of the MDSC is unique for this reason. In some cases, the MDSC will have to determine and enforce optical bypass to find a feasible reroute path upon packet core network failure which cannot be resolved the packet core network itself.

To coordinate this operation, the PNCs will need to update its domain level abstract topology upon resource changes due to a network failure or other factors. The MDSC will incorporate all these update to determine if an alternative E2E reroute path is necessary or not based on the changes reported from the PNCs. 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 also need to perform the path restoration signaling to the affected PNCs whenever necessary.

PCE Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2017

D. Dhody, Ed.
Huawei Technologies
J. Hardwick
Metaswitch
V. Beeram
Juniper Networks
J. Tantsura
July 7, 2016

A YANG Data Model for Path Computation Element Communications Protocol (PCEP)
draft-pkd-pce-pcep-yang-06

Abstract

This document defines a YANG data model for the management of Path Computation Element communications Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs. The data model includes configuration data and state data (status information and counters for the collection of statistics).

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 8, 2017.

Copyright Notice

Table of Contents

1. Introduction

The Path Computation Element (PCE) defined in [RFC4655] is an entity that is capable of computing a network path or route based on a network graph, and applying computational constraints. A Path

Computation Client (PCC) may make requests to a PCE for paths to be computed.

PCEP is the communication protocol between a PCC and PCE and is defined in [RFC5440]. PCEP interactions include path computation requests and path computation replies as well as notifications of specific states related to the use of a PCE in the context of Multiprotocol Label Switching (MPLS) and Generalized MPLS (GMPLS) Traffic Engineering (TE). [I-D.ietf-pce-stateful-pce] specifies extensions to PCEP to enable stateful control of MPLS TE LSPs.

This document defines a YANG [RFC6020] data model for the management of PCEP speakers. It is important to establish a common data model for how PCEP speakers are identified, configured, and monitored. The data model includes configuration data and state data (status information and counters for the collection of statistics).

This document contains a specification of the PCEP YANG module, "ietf-pcep" which provides the PCEP [RFC5440] data model.

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 [RFC2119].

3. Terminology and Notation

This document uses the terminology defined in [RFC4655] and [RFC5440]. In particular, it uses the following acronyms.

- Path Computation Request message (PCReq).
- Path Computation Reply message (PCRep).
- Notification message (PCNtf).
- Error message (PCErr).
- Request Parameters object (RP).
- Synchronization Vector object (SVEC).
- Explicit Route object (ERO).

This document also uses the following terms defined in [RFC7420]:

- PCEP entity: a local PCEP speaker.

- PCEP peer: to refer to a remote PCEP speaker.
- PCEP speaker: where it is not necessary to distinguish between local and remote.

Further, this document also uses the following terms defined in [I-D.ietf-pce-stateful-pce] :

- Stateful PCE, Passive Stateful PCE, Active Stateful PCE
- Delegation, Revocation, Redelegation
- LSP State Report, Path Computation Report message (PCRpt).
- LSP State Update, Path Computation Update message (PCUpd).

[I-D.ietf-pce-pce-initiated-lsp] :

- PCE-initiated LSP, Path Computation LSP Initiate Message (PCInitiate).

[I-D.ietf-pce-lsp-setup-type] :

- Path Setup Type (PST).

[I-D.ietf-pce-segment-routing] :

- Segment Routing (SR).
- Segment Identifier (SID).
- Maximum SID Depth (MSD).

3.1. Tree Diagrams

A graphical representation of the complete data tree is presented in Section 5. The meaning of the symbols in these diagrams is as follows and as per [I-D.ietf-netmod-rfc6087bis]:

- Brackets "[" and "]" enclose list keys.
- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" state data (read-only).

- Symbols after data node names: "?" means an optional node and "*" denotes a "list" or "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

3.2. Prefixes in Data Node Names

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

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and corresponding YANG modules

4. Objectives

This section describes some of the design objectives for the model:

- In case of existing implementations, it needs to map the data model defined in this document to their proprietary native data model. To facilitate such mappings, the data model should be simple.
- The data model should be suitable for new implementations to use as is.
- Mapping to the PCEP MIB Module should be clear.
- The data model should allow for static configurations of peers.
- The data model should include read-only counters in order to gather statistics for sent and received PCEP messages, received messages with errors, and messages that could not be sent due to errors.

   o It should be fairly straightforward to augment the base data model for advanced PCE features.

5. The Design of PCEP Data Model

The module, "ietf-pcep", defines the basic components of a PCE speaker.

```
module: ietf-pcep
   +--rw pcep!
   |  +--rw entity
   |     +--rw addr                          inet:ip-address
   |     +--rw enabled?                      boolean
   |     +--rw role                          pcep-role
   |     +--rw description?                  string
   |     +--rw domain
   |     |  +--rw domain* [domain-type domain]
   |     |     +--rw domain-type    domain-type
   |     |     +--rw domain         domain
   |     +--rw capability
   |     |  +--rw gmpls?                 boolean {gmpls}?
   |     |  +--rw bi-dir?                boolean
   |     |  +--rw diverse?               boolean
   |     |  +--rw load-balance?          boolean
   |     |  +--rw synchronize?           boolean {svec}?
   |     |  +--rw objective-function?    boolean {obj-fn}?
   |     |  +--rw add-path-constraint?   boolean
   |     |  +--rw prioritization?        boolean
   |     |  +--rw multi-request?         boolean
   |     |  +--rw gco?                   boolean {gco}?
   |     |  +--rw p2mp?                  boolean {p2mp}?
   |     |  +--rw stateful {stateful}?
   |     |  |  +--rw enabled?         boolean
   |     |  |  +--rw active?          boolean
   |     |  |  +--rw pce-initiated?   boolean {pce-initiated}?
   |     |  +--rw sr {sr}?
   |     |     +--rw enabled?   boolean
   |     |     +--rw msd?       uint8
   |     +--rw pce-info
   |     |  +--rw scope
   |     |  |  +--rw intra-area-scope?           boolean
   |     |  |  +--rw intra-area-pref?            uint8
   |     |  |  +--rw inter-area-scope?           boolean
   |     |  |  +--rw inter-area-scope-default?   boolean
   |     |  |  +--rw inter-area-pref?            uint8
   |     |  |  +--rw inter-as-scope?             boolean
   |     |  |  +--rw inter-as-scope-default?     boolean
   |     |  |  +--rw inter-as-pref?              uint8
```

```
|     |  |  +--rw inter-layer-scope?          boolean
|     |  |  +--rw inter-layer-pref?           uint8
|     |  +--rw neigh-domains
|     |  |  +--rw domain* [domain-type domain]
|     |  |     +--rw domain-type    domain-type
|     |  |     +--rw domain         domain
|     |  +--rw (auth-type-selection)?
|     |     +--:(auth-key-chain)
|     |     |  +--rw key-chain?          key-chain:key-chain-ref
|     |     +--:(auth-key)
|     |     |  +--rw key?                string
|     |     |  +--rw crypto-algorithm
|     |     |     +--rw (algorithm)?
|     |     |        +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
|     |     |        |  +--rw hmac-sha1-12?             empty
|     |     |        +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
|     |     |        |  +--rw aes-cmac-prf-128?         empty
|     |     |        +--:(md5)
|     |     |        |  +--rw md5?                      empty
|     |     |        +--:(sha-1)
|     |     |        |  +--rw sha-1?                    empty
|     |     |        +--:(hmac-sha-1)
|     |     |        |  +--rw hmac-sha-1?               empty
|     |     |        +--:(hmac-sha-256)
|     |     |        |  +--rw hmac-sha-256?             empty
|     |     |        +--:(hmac-sha-384)
|     |     |        |  +--rw hmac-sha-384?             empty
|     |     |        +--:(hmac-sha-512)
|     |     |        |  +--rw hmac-sha-512?             empty
|     |     |        +--:(clear-text) {clear-text}?
|     |     |        |  +--rw clear-text?               empty
|     |     |        +--:(replay-protection-only) {replay-protection-only}?
|     |     |           +--rw replay-protection-only?   empty
|     |     +--:(auth-tls) {tls}?
|     |        +--rw tls
|     +--rw connect-timer?               uint32
|     +--rw connect-max-retry?           uint32
|     +--rw init-backoff-timer?          uint32
|     +--rw max-backoff-timer?           uint32
|     +--rw open-wait-timer?             uint32
|     +--rw keep-wait-timer?             uint32
|     +--rw keep-alive-timer?            uint32
|     +--rw dead-timer?                  uint32
|     +--rw allow-negotiation?           boolean
|     +--rw max-keep-alive-timer?        uint32
|     +--rw max-dead-timer?              uint32
|     +--rw min-keep-alive-timer?        uint32
|     +--rw min-dead-timer?              uint32
```

```
|     +--rw sync-timer?                   uint32 {svec}?
|     +--rw request-timer?                uint32
|     +--rw max-sessions?                 uint32
|     +--rw max-unknown-reqs?             uint32
|     +--rw max-unknown-msgs?             uint32
|     +--rw pcep-notification-max-rate    uint32
|     +--rw stateful-parameter {stateful}?
|     |  +--rw state-timeout?          uint32
|     |  +--rw redelegation-timeout?   uint32
|     |  +--rw rpt-non-pcep-lsp?       boolean
|     +--rw peers
|        +--rw peer* [addr]
|           +--rw addr                inet:ip-address
|           +--rw description?        string
|           +--rw domain
|           |  +--rw domain* [domain-type domain]
|           |     +--rw domain-type    domain-type
|           |     +--rw domain         domain
|           +--rw capability
|           |  +--rw gmpls?                 boolean {gmpls}?
|           |  +--rw bi-dir?                boolean
|           |  +--rw diverse?               boolean
|           |  +--rw load-balance?          boolean
|           |  +--rw synchronize?           boolean {svec}?
|           |  +--rw objective-function?    boolean {obj-fn}?
|           |  +--rw add-path-constraint?   boolean
|           |  +--rw prioritization?        boolean
|           |  +--rw multi-request?         boolean
|           |  +--rw gco?                   boolean {gco}?
|           |  +--rw p2mp?                  boolean {p2mp}?
|           |  +--rw stateful {stateful}?
|           |  |  +--rw enabled?         boolean
|           |  |  +--rw active?          boolean
|           |  |  +--rw pce-initiated?   boolean {pce-initiated}?
|           |  +--rw sr {sr}?
|           |     +--rw enabled?   boolean
|           |     +--rw msd?       uint8
|           +--rw scope
|           |  +--rw intra-area-scope?           boolean
|           |  +--rw intra-area-pref?            uint8
|           |  +--rw inter-area-scope?           boolean
|           |  +--rw inter-area-scope-default?   boolean
|           |  +--rw inter-area-pref?            uint8
|           |  +--rw inter-as-scope?             boolean
|           |  +--rw inter-as-scope-default?     boolean
|           |  +--rw inter-as-pref?              uint8
|           |  +--rw inter-layer-scope?          boolean
|           |  +--rw inter-layer-pref?           uint8
```

```
   |           +--rw neigh-domains
   |           |  +--rw domain* [domain-type domain]
   |           |     +--rw domain-type    domain-type
   |           |     +--rw domain         domain
   |           +--rw delegation-pref?    uint8 {stateful}?
   |           +--rw (auth-type-selection)?
   |              +--:(auth-key-chain)
   |              |  +--rw key-chain?          key-chain:key-chain-ref
   |              +--:(auth-key)
   |              |  +--rw key?                string
   |              |  +--rw crypto-algorithm
   |              |     +--rw (algorithm)?
   |              |        +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
   |              |        |  +--rw hmac-sha1-12?             empty
   |              |        +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
   |              |        |  +--rw aes-cmac-prf-128?         empty
   |              |        +--:(md5)
   |              |        |  +--rw md5?                      empty
   |              |        +--:(sha-1)
   |              |        |  +--rw sha-1?                    empty
   |              |        +--:(hmac-sha-1)
   |              |        |  +--rw hmac-sha-1?               empty
   |              |        +--:(hmac-sha-256)
   |              |        |  +--rw hmac-sha-256?             empty
   |              |        +--:(hmac-sha-384)
   |              |        |  +--rw hmac-sha-384?             empty
   |              |        +--:(hmac-sha-512)
   |              |        |  +--rw hmac-sha-512?             empty
   |              |        +--:(clear-text) {clear-text}?
   |              |        |  +--rw clear-text?               empty
   |              |        +--:(replay-protection-only) {replay-protection-only}
?
   |              |           +--rw replay-protection-only?   empty
   |              +--:(auth-tls) {tls}?
   |                 +--rw tls
   +--ro pcep-state
      +--ro entity
         +--ro addr?                  inet:ip-address
         +--ro index?                 uint32
         +--ro admin-status?          pcep-admin-status
         +--ro oper-status?           pcep-admin-status
         +--ro role?                  pcep-role
         +--ro domain
         |  +--ro domain* [domain-type domain]
         |     +--ro domain-type    domain-type
         |     +--ro domain         domain
         +--ro capability
         |  +--ro gmpls?                boolean {gmpls}?
         |  +--ro bi-dir?               boolean
```

```
|  +--ro diverse?               boolean
|  +--ro load-balance?          boolean
|  +--ro synchronize?           boolean {svec}?
|  +--ro objective-function?    boolean {obj-fn}?
|  +--ro add-path-constraint?   boolean
|  +--ro prioritization?        boolean
|  +--ro multi-request?         boolean
|  +--ro gco?                   boolean {gco}?
|  +--ro p2mp?                  boolean {p2mp}?
|  +--ro stateful {stateful}?
|  |  +--ro enabled?         boolean
|  |  +--ro active?          boolean
|  |  +--ro pce-initiated?   boolean {pce-initiated}?
|  +--ro sr {sr}?
|     +--ro enabled?   boolean
|     +--ro msd?       uint8
+--ro pce-info
|  +--ro scope
|  |  +--ro intra-area-scope?           boolean
|  |  +--ro intra-area-pref?            uint8
|  |  +--ro inter-area-scope?           boolean
|  |  +--ro inter-area-scope-default?   boolean
|  |  +--ro inter-area-pref?            uint8
|  |  +--ro inter-as-scope?             boolean
|  |  +--ro inter-as-scope-default?     boolean
|  |  +--ro inter-as-pref?              uint8
|  |  +--ro inter-layer-scope?          boolean
|  |  +--ro inter-layer-pref?           uint8
|  +--ro neigh-domains
|  |  +--ro domain* [domain-type domain]
|  |     +--ro domain-type    domain-type
|  |     +--ro domain         domain
|  +--ro (auth-type-selection)?
|     +--:(auth-key-chain)
|     |  +--ro key-chain?          key-chain:key-chain-ref
|     +--:(auth-key)
|     |  +--ro key?                string
|     |  +--ro crypto-algorithm
|     |     +--ro (algorithm)?
|     |        +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
|     |        |  +--ro hmac-sha1-12?             empty
|     |        +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
|     |        |  +--ro aes-cmac-prf-128?         empty
|     |        +--:(md5)
|     |        |  +--ro md5?                      empty
|     |        +--:(sha-1)
|     |        |  +--ro sha-1?                    empty
|     |        +--:(hmac-sha-1)
```

```
|     |       |  +--ro hmac-sha-1?              empty
|     |       +--:(hmac-sha-256)
|     |       |  +--ro hmac-sha-256?            empty
|     |       +--:(hmac-sha-384)
|     |       |  +--ro hmac-sha-384?            empty
|     |       +--:(hmac-sha-512)
|     |       |  +--ro hmac-sha-512?            empty
|     |       +--:(clear-text) {clear-text}?
|     |       |  +--ro clear-text?              empty
|     |       +--:(replay-protection-only) {replay-protection-only}?
|     |          +--ro replay-protection-only?  empty
|     +--:(auth-tls) {tls}?
|        +--ro tls
+--ro connect-timer?          uint32
+--ro connect-max-retry?      uint32
+--ro init-backoff-timer?     uint32
+--ro max-backoff-timer?      uint32
+--ro open-wait-timer?        uint32
+--ro keep-wait-timer?        uint32
+--ro keep-alive-timer?       uint32
+--ro dead-timer?             uint32
+--ro allow-negotiation?      boolean
+--ro max-keep-alive-timer?   uint32
+--ro max-dead-timer?         uint32
+--ro min-keep-alive-timer?   uint32
+--ro min-dead-timer?         uint32
+--ro sync-timer?             uint32 {svec}?
+--ro request-timer?          uint32
+--ro max-sessions?           uint32
+--ro max-unknown-reqs?       uint32
+--ro max-unknown-msgs?       uint32
+--ro stateful-parameter {stateful}?
|  +--ro state-timeout?         uint32
|  +--ro redelegation-timeout?  uint32
|  +--ro rpt-non-pcep-lsp?      boolean
+--ro lsp-db {stateful}?
|  +--ro association-list* [id source global-source extended-id]
|  |  +--ro type?            assoc-type
|  |  +--ro id               uint16
|  |  +--ro source           inet:ip-address
|  |  +--ro global-source    uint32
|  |  +--ro extended-id      string
|  |  +--ro lsp* [plsp-id pcc-id]
|  |     +--ro plsp-id    -> /pcep-state/entity/lsp-db/lsp/plsp-id
|  |     +--ro pcc-id     -> /pcep-state/entity/lsp-db/lsp/pcc-id
|  +--ro lsp* [plsp-id pcc-id]
|     +--ro plsp-id              uint32
|     +--ro pcc-id               inet:ip-address
```

```
         |     +--ro lsp-ref
         |     |  +--ro source?               -> /te:te/lsps-state/lsp/source
         |     |  +--ro destination?          -> /te:te/lsps-state/lsp/destinati
on
         |     |  +--ro tunnel-id?            -> /te:te/lsps-state/lsp/tunnel-id
         |     |  +--ro lsp-id?               -> /te:te/lsps-state/lsp/lsp-id
         |     |  +--ro extended-tunnel-id?   -> /te:te/lsps-state/lsp/extended-
tunnel-id
         |     |  +--ro type?                 -> /te:te/lsps-state/lsp/type
         |     +--ro admin-state?          boolean
         |     +--ro operational-state?    operational-state
         |     +--ro delegated
         |     |  +--ro enabled?   boolean
         |     |  +--ro pce?       -> /pcep-state/entity/peers/peer/addr
         |     |  +--ro srp-id?    uint32
         |     +--ro initiation {pce-initiated}?
         |     |  +--ro enabled?   boolean
         |     |  +--ro pce?       -> /pcep-state/entity/peers/peer/addr
         |     +--ro symbolic-path-name?   string
         |     +--ro last-error?           lsp-error
         |     +--ro pst?                  pst
         |     +--ro association-list* [id source global-source extended-id]
         |        +--ro id               -> /pcep-state/entity/lsp-db/associatio
n-list/id
         |        +--ro source           -> /pcep-state/entity/lsp-db/associatio
n-list/source
         |        +--ro global-source    -> /pcep-state/entity/lsp-db/associatio
n-list/global-source
         |        +--ro extended-id      -> /pcep-state/entity/lsp-db/associatio
n-list/extended-id
         +--ro peers
            +--ro peer* [addr]
               +--ro addr                    inet:ip-address
               +--ro role?                   pcep-role
               +--ro domain
               |  +--ro domain* [domain-type domain]
               |     +--ro domain-type    domain-type
               |     +--ro domain         domain
               +--ro capability
               |  +--ro gmpls?                boolean {gmpls}?
               |  +--ro bi-dir?               boolean
               |  +--ro diverse?              boolean
               |  +--ro load-balance?         boolean
               |  +--ro synchronize?          boolean {svec}?
               |  +--ro objective-function?   boolean {obj-fn}?
               |  +--ro add-path-constraint?  boolean
               |  +--ro prioritization?       boolean
               |  +--ro multi-request?        boolean
               |  +--ro gco?                  boolean {gco}?
               |  +--ro p2mp?                 boolean {p2mp}?
               |  +--ro stateful {stateful}?
               |  |  +--ro enabled?         boolean
               |  |  +--ro active?          boolean
               |  |  +--ro pce-initiated?   boolean {pce-initiated}?
```

```
               |  +--ro sr {sr}?
               |     +--ro enabled?   boolean
               |     +--ro msd?       uint8
               +--ro pce-info
               |  +--ro scope
               |  |  +--ro intra-area-scope?           boolean
               |  |  +--ro intra-area-pref?            uint8
               |  |  +--ro inter-area-scope?           boolean
               |  |  +--ro inter-area-scope-default?   boolean
               |  |  +--ro inter-area-pref?            uint8
               |  |  +--ro inter-as-scope?             boolean
               |  |  +--ro inter-as-scope-default?     boolean
               |  |  +--ro inter-as-pref?              uint8
               |  |  +--ro inter-layer-scope?          boolean
               |  |  +--ro inter-layer-pref?           uint8
               |  +--ro neigh-domains
               |     +--ro domain* [domain-type domain]
               |        +--ro domain-type    domain-type
               |        +--ro domain         domain
               +--ro delegation-pref?        uint8 {stateful}?
               +--ro (auth-type-selection)?
               |  +--:(auth-key-chain)
               |  |  +--ro key-chain?              key-chain:key-chain-ref
               |  +--:(auth-key)
               |  |  +--ro key?                    string
               |  |  +--ro crypto-algorithm
               |  |     +--ro (algorithm)?
               |  |        +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
               |  |        |  +--ro hmac-sha1-12?             empty
               |  |        +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
               |  |        |  +--ro aes-cmac-prf-128?         empty
               |  |        +--:(md5)
               |  |        |  +--ro md5?                     empty
               |  |        +--:(sha-1)
               |  |        |  +--ro sha-1?                   empty
               |  |        +--:(hmac-sha-1)
               |  |        |  +--ro hmac-sha-1?              empty
               |  |        +--:(hmac-sha-256)
               |  |        |  +--ro hmac-sha-256?            empty
               |  |        +--:(hmac-sha-384)
               |  |        |  +--ro hmac-sha-384?            empty
               |  |        +--:(hmac-sha-512)
               |  |        |  +--ro hmac-sha-512?            empty
               |  |        +--:(clear-text) {clear-text}?
               |  |        |  +--ro clear-text?              empty
               |  |        +--:(replay-protection-only) {replay-protection-only}
?
               |  |           +--ro replay-protection-only?   empty
               |  +--:(auth-tls) {tls}?
```

```
|     +--ro tls
+--ro discontinuity-time?     yang:timestamp
+--ro initiate-session?       boolean
+--ro session-exists?         boolean
+--ro num-sess-setup-ok?      yang:counter32
+--ro num-sess-setup-fail?    yang:counter32
+--ro session-up-time?        yang:timestamp
+--ro session-fail-time?      yang:timestamp
+--ro session-fail-up-time?   yang:timestamp
+--ro pcep-stats
|  +--ro avg-rsp-time?               uint32
|  +--ro lwm-rsp-time?               uint32
|  +--ro hwm-rsp-time?               uint32
|  +--ro num-pcreq-sent?             yang:counter32
|  +--ro num-pcreq-rcvd?             yang:counter32
|  +--ro num-pcrep-sent?             yang:counter32
|  +--ro num-pcrep-rcvd?             yang:counter32
|  +--ro num-pcerr-sent?             yang:counter32
|  +--ro num-pcerr-rcvd?             yang:counter32
|  +--ro num-pcntf-sent?             yang:counter32
|  +--ro num-pcntf-rcvd?             yang:counter32
|  +--ro num-keepalive-sent?         yang:counter32
|  +--ro num-keepalive-rcvd?         yang:counter32
|  +--ro num-unknown-rcvd?           yang:counter32
|  +--ro num-corrupt-rcvd?           yang:counter32
|  +--ro num-req-sent?               yang:counter32
|  +--ro num-req-sent-pend-rep?      yang:counter32
|  +--ro num-req-sent-ero-rcvd?      yang:counter32
|  +--ro num-req-sent-nopath-rcvd?   yang:counter32
|  +--ro num-req-sent-cancel-rcvd?   yang:counter32
|  +--ro num-req-sent-error-rcvd?    yang:counter32
|  +--ro num-req-sent-timeout?       yang:counter32
|  +--ro num-req-sent-cancel-sent?   yang:counter32
|  +--ro num-req-rcvd?               yang:counter32
|  +--ro num-req-rcvd-pend-rep?      yang:counter32
|  +--ro num-req-rcvd-ero-sent?      yang:counter32
|  +--ro num-req-rcvd-nopath-sent?   yang:counter32
|  +--ro num-req-rcvd-cancel-sent?   yang:counter32
|  +--ro num-req-rcvd-error-sent?    yang:counter32
|  +--ro num-req-rcvd-cancel-rcvd?   yang:counter32
|  +--ro num-rep-rcvd-unknown?       yang:counter32
|  +--ro num-req-rcvd-unknown?       yang:counter32
|  +--ro svec {svec}?
|  |  +--ro num-svec-sent?       yang:counter32
|  |  +--ro num-svec-req-sent?   yang:counter32
|  |  +--ro num-svec-rcvd?       yang:counter32
|  |  +--ro num-svec-req-rcvd?   yang:counter32
|  +--ro stateful {stateful}?
```

```
|  |  +--ro num-pcrpt-sent?             yang:counter32
|  |  +--ro num-pcrpt-rcvd?             yang:counter32
|  |  +--ro num-pcupd-sent?             yang:counter32
|  |  +--ro num-pcupd-rcvd?             yang:counter32
|  |  +--ro num-rpt-sent?               yang:counter32
|  |  +--ro num-rpt-rcvd?               yang:counter32
|  |  +--ro num-rpt-rcvd-error-sent?    yang:counter32
|  |  +--ro num-upd-sent?               yang:counter32
|  |  +--ro num-upd-rcvd?               yang:counter32
|  |  +--ro num-upd-rcvd-unknown?       yang:counter32
|  |  +--ro num-upd-rcvd-undelegated?   yang:counter32
|  |  +--ro num-upd-rcvd-error-sent?    yang:counter32
|  |  +--ro initiation {pce-initiated}?
|  |     +--ro num-pcinitiate-sent?            yang:counter32
|  |     +--ro num-pcinitiate-rcvd?            yang:counter32
|  |     +--ro num-initiate-sent?              yang:counter32
|  |     +--ro num-initiate-rcvd?              yang:counter32
|  |     +--ro num-initiate-rcvd-error-sent?   yang:counter32
|  +--ro num-req-sent-closed?        yang:counter32
|  +--ro num-req-rcvd-closed?        yang:counter32
+--ro sessions
   +--ro session* [initiator]
      +--ro initiator               pcep-initiator
      +--ro state-last-change?      yang:timestamp
      +--ro state?                  pcep-sess-state
      +--ro session-creation?       yang:timestamp
      +--ro connect-retry?          yang:counter32
      +--ro local-id?               uint32
      +--ro remote-id?              uint32
      +--ro keepalive-timer?        uint32
      +--ro peer-keepalive-timer?   uint32
      +--ro dead-timer?             uint32
      +--ro peer-dead-timer?        uint32
      +--ro ka-hold-time-rem?       uint32
      +--ro overloaded?             boolean
      +--ro overload-time?          uint32
      +--ro peer-overloaded?        boolean
      +--ro peer-overload-time?     uint32
      +--ro lspdb-sync?             sync-state {stateful}?
      +--ro discontinuity-time?     yang:timestamp
      +--ro pcep-stats
         +--ro avg-rsp-time?              uint32
         +--ro lwm-rsp-time?              uint32
         +--ro hwm-rsp-time?              uint32
         +--ro num-pcreq-sent?            yang:counter32
         +--ro num-pcreq-rcvd?            yang:counter32
         +--ro num-pcrep-sent?            yang:counter32
         +--ro num-pcrep-rcvd?            yang:counter32
```

```
                        +--ro num-pcerr-sent?             yang:counter32
                        +--ro num-pcerr-rcvd?             yang:counter32
                        +--ro num-pcntf-sent?             yang:counter32
                        +--ro num-pcntf-rcvd?             yang:counter32
                        +--ro num-keepalive-sent?         yang:counter32
                        +--ro num-keepalive-rcvd?         yang:counter32
                        +--ro num-unknown-rcvd?           yang:counter32
                        +--ro num-corrupt-rcvd?           yang:counter32
                        +--ro num-req-sent?               yang:counter32
                        +--ro num-req-sent-pend-rep?      yang:counter32
                        +--ro num-req-sent-ero-rcvd?      yang:counter32
                        +--ro num-req-sent-nopath-rcvd?   yang:counter32
                        +--ro num-req-sent-cancel-rcvd?   yang:counter32
                        +--ro num-req-sent-error-rcvd?    yang:counter32
                        +--ro num-req-sent-timeout?       yang:counter32
                        +--ro num-req-sent-cancel-sent?   yang:counter32
                        +--ro num-req-rcvd?               yang:counter32
                        +--ro num-req-rcvd-pend-rep?      yang:counter32
                        +--ro num-req-rcvd-ero-sent?      yang:counter32
                        +--ro num-req-rcvd-nopath-sent?   yang:counter32
                        +--ro num-req-rcvd-cancel-sent?   yang:counter32
                        +--ro num-req-rcvd-error-sent?    yang:counter32
                        +--ro num-req-rcvd-cancel-rcvd?   yang:counter32
                        +--ro num-rep-rcvd-unknown?       yang:counter32
                        +--ro num-req-rcvd-unknown?       yang:counter32
                        +--ro svec {svec}?
                        |  +--ro num-svec-sent?       yang:counter32
                        |  +--ro num-svec-req-sent?   yang:counter32
                        |  +--ro num-svec-rcvd?       yang:counter32
                        |  +--ro num-svec-req-rcvd?   yang:counter32
                        +--ro stateful {stateful}?
                           +--ro num-pcrpt-sent?             yang:counter32
                           +--ro num-pcrpt-rcvd?             yang:counter32
                           +--ro num-pcupd-sent?             yang:counter32
                           +--ro num-pcupd-rcvd?             yang:counter32
                           +--ro num-rpt-sent?               yang:counter32
                           +--ro num-rpt-rcvd?               yang:counter32
                           +--ro num-rpt-rcvd-error-sent?    yang:counter32
                           +--ro num-upd-sent?               yang:counter32
                           +--ro num-upd-rcvd?               yang:counter32
                           +--ro num-upd-rcvd-unknown?       yang:counter32
                           +--ro num-upd-rcvd-undelegated?   yang:counter32
                           +--ro num-upd-rcvd-error-sent?    yang:counter32
                           +--ro initiation {pce-initiated}?
                              +--ro num-pcinitiate-sent?           yang:counter
32
                              +--ro num-pcinitiate-rcvd?           yang:counter
32
                              +--ro num-initiate-sent?             yang:counter
32
                              +--ro num-initiate-rcvd?             yang:counter
32
```

```
                              +--ro num-initiate-rcvd-error-sent?   yang:counter
32
notifications:
   +---n pcep-session-up
   |  +--ro peer-addr?           -> /pcep-state/entity/peers/peer/addr
   |  +--ro session-initiator?   -> /pcep-state/entity/peers/peer/sessions/sessi
on/initiator
   |  +--ro state-last-change?   yang:timestamp
   |  +--ro state?               pcep-sess-state
   +---n pcep-session-down
   |  +--ro peer-addr?           -> /pcep-state/entity/peers/peer/addr
   |  +--ro session-initiator?   pcep-initiator
   |  +--ro state-last-change?   yang:timestamp
   |  +--ro state?               pcep-sess-state
   +---n pcep-session-local-overload
   |  +--ro peer-addr?           -> /pcep-state/entity/peers/peer/addr
   |  +--ro session-initiator?   -> /pcep-state/entity/peers/peer/sessions/sessi
on/initiator
   |  +--ro overloaded?          boolean
   |  +--ro overload-time?       uint32
   +---n pcep-session-local-overload-clear
   |  +--ro peer-addr?    -> /pcep-state/entity/peers/peer/addr
   |  +--ro overloaded?   boolean
   +---n pcep-session-peer-overload
   |  +--ro peer-addr?            -> /pcep-state/entity/peers/peer/addr
   |  +--ro session-initiator?    -> /pcep-state/entity/peers/peer/sessions/sess
ion/initiator
   |  +--ro peer-overloaded?      boolean
   |  +--ro peer-overload-time?   uint32
   +---n pcep-session-peer-overload-clear
      +--ro peer-addr?         -> /pcep-state/entity/peers/peer/addr
      +--ro peer-overloaded?   boolean
```

5.1. The Entity

The PCEP yang module may contain status information for the local PCEP entity.

The entity has an IP address (using ietf-inet-types [RFC6991]) and a "role" leaf (the local entity PCEP role) as mandatory.

Note that, the PCEP MIB module [RFC7420] uses an entity list and a system generated entity index as a primary index to the read only entity table. If the device implements the PCEP MIB, the "index" leaf MUST contain the value of the corresponding pcePcepEntityIndex and only one entity is assumed.

5.2. The Peer Lists

The peer list contains peer(s) that the local PCEP entity knows about. A PCEP speaker is identified by its IP address. If there is a PCEP speaker in the network that uses multiple IP addresses then it looks like multiple distinct peers to the other PCEP speakers in the network.

Since PCEP sessions can be ephemeral, the peer list tracks a peer even when no PCEP session currently exists to that peer. The statistics contained are an aggregate of the statistics for all successive sessions to that peer.

To limit the quantity of information that is stored, an implementation MAY choose to discard this information if and only if no PCEP session exists to the corresponding peer.

The data model for PCEP peer presented in this document uses a flat list of peers. Each peer in the list is identified by its IP address (addr-type, addr).

There is one list for static peer configuration ("/pcep/entity/peers"), and a separate list for the operational state of all peers (i.e. static as well as discovered)("/pcep-state/entity/peers"). The former is used to enable remote PCE configuration at PCC (or PCE) while the latter has the operational state of these peers as well as the remote PCE peer which were discovered and PCC peers that have initiated session.

5.3. The Session Lists

The session list contains PCEP session that the PCEP entity (PCE or PCC) is currently participating in. The statistics in session are semantically different from those in peer since the former applies to the current session only, whereas the latter is the aggregate for all sessions that have existed to that peer.

Although [RFC5440] forbids more than one active PCEP session between a given pair of PCEP entities at any given time, there is a window during session establishment where two sessions may exist for a given pair, one representing a session initiated by the local PCEP entity and the other representing a session initiated by the peer. If either of these sessions reaches active state first, then the other is discarded.

The data model for PCEP session presented in this document uses a flat list of sessions. Each session in the list is identified by its

initiator. This index allows two sessions to exist transiently for a given peer, as discussed above.

There is only one list for the operational state of all sessions ("/pcep-state/entity/peers/peer/sessions/session").

5.4. Notifications

This YANG model defines a list of notifications to inform client of important events detected during the protocol operation. The notifications defined cover the PCEP MIB notifications.

6. Advanced PCE Features

This document contains a specification of the base PCEP YANG module, "ietf-pcep" which provides the basic PCEP [RFC5440] data model.

This document further handles advanced PCE features like -

- Capability and Scope
- Domain information (local/neighbour)
- Path-Key
- OF
- GCO
- P2MP
- GMPLS
- Inter-Layer
- Stateful PCE
- Segement Routing
- Authentication including PCEPS (TLS)

[Editor’s Note - Some of them would be added in a future revision.]

6.1. Stateful PCE’s LSP-DB

In the operational state of PCEP which supports stateful PCE mode, the list of LSP state are maintained in LSP-DB. The key is the PLSP-ID and the PCC IP address.

The PCEP data model contains the operational state of LSPs (/pcep-state/entity/lsp-db/lsp/) with PCEP specific attributes. The generic TE attributes of the LSP are defined in [I-D.ietf-teas-yang-te]. A reference to LSP state in TE model is maintained.

7. Open Issues and Next Step

This section is added so that open issues can be tracked. This section would be removed when the document is ready for publication.

7.1. The PCE-Initiated LSP

The TE Model at [I-D.ietf-teas-yang-te] should support creationg of tunnels at the controller (PCE) and marking them as PCE-Initiated. The LSP-DB in the PCEP Yang (/pcep-state/entity/lsp-db/lsp/initiation) also marks the LSPs which are PCE-initiated.

7.2. PCEP over TLS (PCEPS)

A future version of this document would add TLS related configurations.

8. PCEP YANG Module

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

```
<CODE BEGINS> file "ietf-pcep@2016-07-07.yang"
module ietf-pcep {
    namespace "urn:ietf:params:xml:ns:yang:ietf-pcep";
    prefix pcep;

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

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

    import ietf-te {
        prefix "te";
    }

    import ietf-key-chain {
       prefix "key-chain";
```

```
    }


    organization
        "IETF PCE (Path Computation Element) Working Group";

    contact
        "WG Web:   <http://tools.ietf.org/wg/pce/>
         WG List:  <mailto:pce@ietf.org>
         WG Chair: JP Vasseur
                   <mailto:jpv@cisco.com>
         WG Chair: Julien Meuric
                   <mailto:julien.meuric@orange.com>
         WG Chair: Jonathan Hardwick
                   <mailto:Jonathan.Hardwick@metaswitch.com>
         Editor:   Dhruv Dhody
                   <mailto:dhruv.ietf@gmail.com>";

    description
        "The YANG module defines a generic configuration and
         operational model for PCEP common across all of the
         vendor implementations.";


    revision 2016-07-07 {
        description "Initial revision.";
        reference
            "RFC XXXX:  A YANG Data Model for Path Computation
                        Element Communications Protocol
                        (PCEP)";
    }

    /*
     * Identities
     */

    identity pcep {
        description "Identity for the PCEP protocol.";
    }

    /*
     * Typedefs
     */
    typedef pcep-role {
        type enumeration {
            enum unknown {
                value "0";
                description
```

```
                "An unknown role";
            }
            enum pcc {
                value "1";
                description
                "The role of a Path Computation Client";
            }
            enum pce {
                value "2";
                description
                "The role of Path Computation Element";
            }
            enum pcc-and-pce {
                value "3";
                description
                "The role of both Path Computation Client and
                 Path Computation Element";
            }
        }

        description
            "The role of a PCEP speaker.
             Takes one of the following values
             - unknown(0): the role is not known.
             - pcc(1): the role is of a Path Computation
               Client (PCC).
             - pce(2): the role is of a Path Computation
               Server (PCE).
             - pccAndPce(3): the role is of both a PCC and
               a PCE.";

    }

    typedef pcep-admin-status {
        type enumeration {
                enum admin-status-up {
                value "1";
                description
                "Admin Status is Up";
            }
            enum admin-status-down {
                value "2";
                description
                "Admin Status is Down";
            }
        }

        description
```

```
        "The Admin Status of the PCEP entity.
         Takes one of the following values
             - admin-status-up(1): Admin Status is Up.
             - admin-status-down(2): Admin Status is Down";
    }

    typedef pcep-oper-status {
        type enumeration {
            enum oper-status-up {
                value "1";
                description
                "The PCEP entity is active";
            }
            enum oper-status-down {
                value "2";
                description
                "The PCEP entity is inactive";
            }
            enum oper-status-going-up {
                value "3";
                description
                "The PCEP entity is activating";
            }
            enum oper-status-going-down {
                value "4";
                description
                "The PCEP entity is deactivating";
            }
            enum oper-status-failed {
                value "5";
                description
                "The PCEP entity has failed and will recover
                 when possible.";
            }
            enum oper-status-failed-perm {
                value "6";
                description
                "The PCEP entity has failed and will not recover
                 without operator intervention";
            }
        }
        description
        "The operational status of the PCEP entity.
         Takes one of the following values
             - oper-status-up(1): Active
             - oper-status-down(2): Inactive
             - oper-status-going-up(3): Activating
             - oper-status-going-down(4): Deactivating
```

```
             - oper-status-failed(5): Failed
             - oper-status-failed-perm(6): Failed Permanantly";
    }

    typedef pcep-initiator {
        type enumeration {
            enum local {
                value "1";
                description
                "The local PCEP entity initiated the session";
            }

            enum remote {
                value "2";
                description
                "The remote PCEP peer initiated the session";
            }
        }
        description
        "The initiator of the session, that is, whether the TCP
         connection was initiated by the local PCEP entity or
         the remote peer.
         Takes one of the following values
             - local(1): Initiated locally
             - remote(2): Initiated remotely";
    }

    typedef pcep-sess-state {
        type enumeration {
            enum tcp-pending {
                value "1";
                description
                    "The tcp-pending state of PCEP session.";
            }

            enum open-wait {
                value "2";
                description
                    "The open-wait state of PCEP session.";
            }

            enum keep-wait {
                value "3";
                description
                    "The keep-wait state of PCEP session.";
            }

            enum session-up {
```

```
                value "4";
                description
                    "The session-up state of PCEP session.";
            }
        }
        description
            "The current state of the session.
             The set of possible states excludes the idle state
             since entries do not exist in the idle state.
             Takes one of the following values
                - tcp-pending(1): PCEP TCP Pending state
                - open-wait(2): PCEP Open Wait state
                - keep-wait(3): PCEP Keep Wait state
                - session-up(4): PCEP Session Up state";
    }

    typedef domain-type {
        type enumeration {
            enum ospf-area {
                value "1";
                description
                    "The OSPF area.";
            }
            enum isis-area {
                value "2";
                description
                    "The IS-IS area.";
            }
            enum as {
                value "3";
                description
                    "The Autonomous System (AS).";
            }
        }
        description
            "The PCE Domain Type";
    }

    typedef domain-ospf-area {
        type union {
            type uint32;
            type yang:dotted-quad;
       }
       description
            "OSPF Area ID.";
    }

    typedef domain-isis-area {
```

```
        type string {
            pattern '[0-9A-Fa-f]{2}\.([0-9A-Fa-f]{4}\.){0,3}';
        }
        description
            "IS-IS Area ID.";
    }

    typedef domain-as {
        type uint32;
        description
            "Autonomous System number.";

    }

    typedef domain {
        type union {
            type domain-ospf-area;
            type domain-isis-area;
            type domain-as;
        }
        description
            "The Domain Information";
    }

    typedef operational-state {
        type enumeration {
            enum down {
                value "0";
                description
                    "not active.";
            }
            enum up {
                value "1";
                description
                    "signalled.";
            }
            enum active {
                value "2";
                description
                    "up and carrying traffic.";
            }
            enum going-down {
                value "3";
                description
                    "LSP is being torn down, resources are
                     being released.";
            }
            enum going-up {
```

```
                value "4";
                description
                    "LSP is being signalled.";
            }
        }
        description
            "The operational status of the LSP";
    }

    typedef lsp-error {
        type enumeration {
            enum no-error {
                value "0";
                description
                    "No error, LSP is fine.";
            }
            enum unknown {
                value "1";
                description
                    "Unknown reason.";
            }
            enum limit {
                value "2";
                description
                    "Limit reached for PCE-controlled LSPs.";
            }
            enum pending {
                value "3";
                description
                    "Too many pending LSP update requests.";
            }
            enum unacceptable {
                value "4";
                description
                    "Unacceptable parameters.";
            }
            enum internal {
                value "5";
                description
                    "Internal error.";
            }
            enum admin {
                value "6";
                description
                    "LSP administratively brought down.";
            }
            enum preempted {
                value "7";
```

```
                description
                    "LSP preempted.";
            }
            enum rsvp {
                value "8";
                description
                    "RSVP signaling error.";
            }
        }
        description
            "The LSP Error Codes.";
    }

    typedef sync-state {
        type enumeration {
            enum pending {
                value "0";
                description
                    "The state synchronization
                     has not started.";
            }
            enum ongoing {
                value "1";
                description
                    "The state synchronization
                     is ongoing.";
            }
            enum finished {
                value "2";
                description
                    "The state synchronization
                     is finished.";
            }
        }
        description
            "The LSP-DB state synchronization operational status.";
    }

        typedef pst{
                type enumeration{
                        enum rsvp-te{
                                value "0";
                                description
                                        "RSVP-TE signaling protocol";
                        }
                        enum sr{
                                value "1";
                                description
```

```
                                        "Segment Routing Traffic Engineering";
                        }
                }
                description
                        "The Path Setup Type";
        }

    typedef assoc-type{
        type enumeration{
            enum protection{
                value "1";
                description
                    "Path Protection Association Type";
            }
        }
        description
            "The PCEP Association Type";
    }

    /*
     * Features
     */

    feature svec {
        description
            "Support synchronized path computation.";
    }

    feature gmpls {
        description
            "Support GMPLS.";
    }

    feature obj-fn {
        description
            "Support OF as per RFC 5541.";
    }

    feature gco {
        description
            "Support GCO as per RFC 5557.";
    }

    feature pathkey {
        description
            "Support pathkey as per RFC 5520.";
    }
```

```
    feature p2mp {
        description
            "Support P2MP as per RFC 6006.";
    }

    feature stateful {
        description
            "Support stateful PCE.";
    }

    feature pce-initiated {
        description
            "Support PCE-Initiated LSP.";
    }

    feature tls {
        description
            "Support PCEP over TLS.";
    }

    feature sr {
        description
            "Support Segement Routing for PCE.";
    }

    /*
     * Groupings
     */



    grouping pcep-entity-info{
        description
            "This grouping defines the attributes for PCEP entity.";
        leaf connect-timer {
            type uint32 {
                range "1..65535";
            }
            units "seconds";
            default 60;
            description
                "The time in seconds that the PCEP entity will wait
                 to establish a TCP connection with a peer.  If a
                 TCP connection is not established within this time
                 then PCEP aborts the session setup attempt.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
```

```
        }

        leaf connect-max-retry {
            type uint32;
            default 5;
            description
                "The maximum number of times the system tries to
                 establish a TCP connection to a peer before the
                 session with the peer transitions to the idle
                 state.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf init-backoff-timer {
            type uint32 {
                range "1..65535";
            }
            units "seconds";
            description
               "The initial back-off time in seconds for retrying
                a failed session setup attempt to a peer.
                The back-off time increases for each failed
                session setup attempt, until a maximum back-off
                time is reached.  The maximum back-off time is
                max-backoff-timer.";
        }

        leaf max-backoff-timer {
            type uint32;
            units "seconds";
            description
               "The maximum back-off time in seconds for retrying
                a failed session setup attempt to a peer.
                The back-off time increases for each failed session
                setup attempt, until this maximum value is reached.
                Session setup attempts then repeat periodically
                without any further increase in back-off time.";
        }

        leaf open-wait-timer {
            type uint32 {
                range "1..65535";
            }
            units "seconds";
            default 60;
            description
```

```
               "The time in seconds that the PCEP entity will wait
                to receive an Open message from a peer after the
                TCP connection has come up.
                If no Open message is received within this time then
                PCEP terminates the TCP connection and deletes the
                associated sessions.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf keep-wait-timer {
            type uint32 {
                range "1..65535";
            }
            units "seconds";
            default 60;
            description
                "The time in seconds that the PCEP entity will wait
                 to receive a Keepalive or PCErr message from a peer
                 during session initialization after receiving an
                 Open message.  If no Keepalive or PCErr message is
                 received within this time then PCEP terminates the
                 TCP connection and deletes the associated
                 sessions.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf keep-alive-timer {
            type uint32 {
                range "0..255";
            }
            units "seconds";
            default 30;
            description
                "The keep alive transmission timer that this PCEP
                 entity will propose in the initial OPEN message of
                 each session it is involved in.  This is the
                 maximum time between two consecutive messages sent
                 to a peer. Zero means that the PCEP entity prefers
                 not to send Keepalives at all.
                 Note that the actual Keepalive transmission
                 intervals, in either direction of an active PCEP
                 session, are determined by negotiation between the
                 peers as specified by RFC 5440, and so may differ
                 from this configured value.";
```

```
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf dead-timer {
            type uint32 {
                range "0..255";
            }
            units "seconds";
            must ". >= ../keep-alive-timer" {
                error-message "The dead timer must be "
                       + "larger than the keep alive timer";
                description
                    "This value MUST be greater than
                     keep-alive-timer.";

            }
            default 120;
            description
                "The dead timer that this PCEP entity will propose
                 in the initial OPEN message of each session it is
                 involved in. This is the time after which a peer
                 should declare a session down if it does not
                 receive any PCEP messages. Zero suggests that the
                 peer does not run a dead timer at all." ;
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf allow-negotiation{
            type boolean;
            description
                "Whether the PCEP entity will permit negotiation of
                 session parameters.";
        }

        leaf max-keep-alive-timer{
            type uint32 {
                range "0..255";
            }
            units "seconds";
            description
                "In PCEP session parameter negotiation in seconds,
                 the maximum value that this PCEP entity will
                 accept from a peer for the interval between
                 Keepalive transmissions. Zero means that the PCEP
```

```
                entity will allow no Keepalive transmission at
                all." ;
        }

        leaf max-dead-timer{
            type uint32 {
                range "0..255";
            }
            units "seconds";
            description
                "In PCEP session parameter negotiation in seconds,
                 the maximum value that this PCEP entity will accept
                 from a peer for the Dead timer.  Zero means that
                 the PCEP entity will allow not running a Dead
                 timer.";
        }

        leaf min-keep-alive-timer{
            type uint32 {
                range "0..255";
            }
            units "seconds";
            description
                "In PCEP session parameter negotiation in seconds,
                 the minimum value that this PCEP entity will
                 accept for the interval between Keepalive
                 transmissions. Zero means that the PCEP entity
                 insists on no Keepalive transmission at all.";
        }

        leaf min-dead-timer{
            type uint32 {
                range "0..255";
            }
            units "seconds";
            description
                 "In PCEP session parameter negotiation in
                  seconds, the minimum value that this PCEP entity
                  will accept for the Dead timer.  Zero means that
                  the PCEP entity insists on not running a Dead
                  timer.";
        }

        leaf sync-timer{
            if-feature svec;
            type uint32 {
                range "0..65535";
            }
```

```
            units "seconds";
            default 60;
            description
                "The value of SyncTimer in seconds is used in the
                 case of synchronized path computation request
                 using the SVEC object. Consider the case where a
                 PCReq message is received by a PCE that contains
                 the SVEC object referring to M synchronized path
                 computation requests.  If after the expiration of
                 the SyncTimer all the M path computation requests
                 have not been, received a protocol error is
                 triggered and the PCE MUST cancel the whole set
                 of path computation  requests.
                 The aim of the SyncTimer is to avoid the storage
                 of unused synchronized requests should one of
                 them get lost for some reasons (for example, a
                 misbehaving PCC).
                 Zero means that the PCEP entity does not use the
                 SyncTimer.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }


        leaf request-timer{
            type uint32 {
                range "1..65535";
            }
            units "seconds";
            description
                "The maximum time that the PCEP entity will wait
                 for a response to a PCReq message.";
        }

        leaf max-sessions{
            type uint32;
            description
               "Maximum number of sessions involving this PCEP
                entity that can exist at any time.";
        }

        leaf max-unknown-reqs{
            type uint32;
            default 5;
            description
              "The maximum number of unrecognized requests and
               replies that any session on this PCEP entity is
```

```
                willing to accept per minute before terminating
                the session.
                A PCRep message contains an unrecognized reply
                if it contains an RP object whose request ID
                does not correspond to any in-progress request
                sent by this PCEP entity.
                A PCReq message contains an unrecognized request
                if it contains an RP object whose request ID is
                zero.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

        leaf max-unknown-msgs{
            type uint32;
            default 5;
            description
             "The maximum number of unknown messages that any
              session on this PCEP entity is willing to accept
              per minute before terminating the session.";
            reference
                "RFC 5440: Path Computation Element (PCE)
                           Communication Protocol (PCEP)";
        }

    }//pcep-entity-info

    grouping  pce-scope{
        description
            "This grouping defines PCE path computation scope
             information which maybe relevant to PCE selection.
             This information corresponds to PCE auto-discovery
             information.";
        reference
            "RFC 5088: OSPF Protocol Extensions for Path
                       Computation Element (PCE)
                       Discovery
             RFC 5089: IS-IS Protocol Extensions for Path
                       Computation Element (PCE)
                       Discovery";
        leaf intra-area-scope{
            type boolean;
            default true;
            description
                "PCE can compute intra-area paths.";
        }
        leaf intra-area-pref{
```

```
            type uint8{
                range "0..7";
            }
            description
              "The PCE's preference for intra-area TE LSP
              computation.";
        }
        leaf inter-area-scope{
            type boolean;
            default false;
            description
                "PCE can compute inter-area paths.";
        }
        leaf inter-area-scope-default{
            type boolean;
            default false;
            description
                "PCE can act as a default PCE for inter-area
                 path computation.";
        }
        leaf inter-area-pref{
            type uint8{
                range "0..7";
            }
            description
              "The PCE's preference for inter-area TE LSP
              computation.";
        }
        leaf inter-as-scope{
            type boolean;
            default false;
            description
                "PCE can compute inter-AS paths.";
        }
        leaf inter-as-scope-default{
            type boolean;
            default false;
            description
                "PCE can act as a default PCE for inter-AS
                 path computation.";
        }
        leaf inter-as-pref{
            type uint8{
                range "0..7";
            }
            description
              "The PCE's preference for inter-AS TE LSP
              computation.";
```

```
        }
        leaf inter-layer-scope{
            type boolean;
            default false;
            description
                "PCE can compute inter-layer paths.";
        }
        leaf inter-layer-pref{
            type uint8{
                range "0..7";
            }
            description
              "The PCE's preference for inter-layer TE LSP
              computation.";
        }
    }//pce-scope

    grouping  domain{
        description
            "This grouping specifies a Domain where the
             PCEP speaker has topology visibility.";
        leaf domain-type{
            type domain-type;
            description
              "The domain type.";
        }
        leaf domain{
            type domain;
            description
              "The domain Information.";
        }
    }//domain

    grouping capability{
        description
            "This grouping specifies a capability
             information of local PCEP entity. This maybe
             relevant to PCE selection as well. This
             information corresponds to PCE auto-discovery
             information.";
        reference
            "RFC 5088: OSPF Protocol Extensions for Path
                       Computation Element (PCE)
                       Discovery
             RFC 5089: IS-IS Protocol Extensions for Path
                       Computation Element (PCE)
                       Discovery";
        leaf gmpls{
```

```
            if-feature gmpls;
            type boolean;
            description
              "Path computation with GMPLS link
               constraints.";
          }
          leaf bi-dir{
            type boolean;
            description
              "Bidirectional path computation.";
          }
          leaf diverse{
            type boolean;
            description
              "Diverse path computation.";
          }
          leaf load-balance{
            type boolean;
            description
              "Load-balanced path computation.";
          }
          leaf synchronize{
            if-feature svec;
            type boolean;
            description
              "Synchronized paths computation.";
          }
          leaf objective-function{
            if-feature obj-fn;
            type boolean;
            description
              "Support for multiple objective functions.";
          }
          leaf add-path-constraint{
            type boolean;
            description
              "Support for additive path constraints (max
               hop count, etc.).";
          }
          leaf prioritization{
            type boolean;
            description
              "Support for request prioritization.";
          }
          leaf multi-request{
            type boolean;
            description
              "Support for multiple requests per message.";
```

```
        }
        leaf gco{
            if-feature gco;
            type boolean;
            description
              "Support for Global Concurrent Optimization
               (GCO).";
        }
        leaf p2mp{
            if-feature p2mp;
            type boolean;
            description
              "Support for P2MP path computation.";
        }

        container stateful{
            if-feature stateful;
            description
                "If stateful PCE feature is present";
            leaf enabled{
                type boolean;
                description
                    "Enabled or Disabled";
            }
            leaf active{
                type boolean;
                description
                  "Support for active stateful PCE.";
            }
            leaf pce-initiated{
                if-feature pce-initiated;
                type boolean;
                description
                  "Support for PCE-initiated LSP.";
            }
        }
                container sr{
                        if-feature sr;
                        description
                                "If segment routing is supported";
                        leaf enabled{
                type boolean;
                description
                    "Enabled or Disabled";
                        }
                        leaf msd{ /*should be in MPLS yang model (?)*/
                                type uint8;
                must "((../../role == 'pcc')" +
```

```
                     " or " +
                    "(../../role == 'pcc-and-pce')))"
                {
                    error-message
                        "The PCEP entity must be PCC";
                    description
                        "When PCEP entity is PCC for
                        MSD to be applicable";
                }
                                description
                                        "Maximum SID Depth";
                        }
                }
    }//capability

    grouping  info{
        description
            "This grouping specifies all information which
             maybe relevant to both PCC and PCE.
             This information corresponds to PCE auto-discovery
             information.";
        container domain{
            description
                "The local domain for the PCEP entity";
            list domain{
                key "domain-type domain";
                description
                    "The local domain.";
                uses domain{
                    description
                        "The local domain for the PCEP entity.";
                }
            }
        }
        container capability{
            description
                "The PCEP entity capability";
            uses capability{
                description
                    "The PCEP entity supported
                    capabilities.";
            }
        }
    }//info

    grouping  pce-info{
        description
            "This grouping specifies all PCE information
```

```
             which maybe relevant to the PCE selection.
             This information corresponds to PCE auto-discovery
             information.";
        container scope{
            description
                "The path computation scope";
            uses pce-scope;
        }

        container neigh-domains{
            description
                "The list of neighbour PCE-Domain
                 toward which a PCE can compute
                 paths";
            list domain{
                key "domain-type domain";

                description
                    "The neighbour domain.";
                uses domain{
                    description
                        "The PCE neighbour domain.";
                }
            }
        }
    }//pce-info

    grouping  pcep-stats{
        description
            "This grouping defines statistics for PCEP. It is used
             for both peer and current session.";
        leaf avg-rsp-time{
            type uint32;
            units "milliseconds";
            must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
                 " or " +
                 "(/pcep-state/entity/peers/peer/role = 'pcc'" +
                 " and avg-rsp-time = 0))" {
                error-message
                    "Invalid average response time";
                description
                    "If role is pcc then this leaf is meaningless
                     and is set to zero.";
            }
            description
              "The average response time.
               If an average response time has not been
               calculated then this leaf has the value zero.";
```

```
        }

        leaf lwm-rsp-time{
            type uint32;
            units "milliseconds";
            must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
                 " or " +
                 "(/pcep-state/entity/peers/peer/role = 'pcc'" +
                 " and lwm-rsp-time = 0))" {
                error-message
                    "Invalid smallest (low-water mark)
                     response time";
                description
                    "If role is pcc then this leaf is meaningless
                     and is set to zero.";
            }
            description
             "The smallest (low-water mark) response time seen.
              If no responses have been received then this
              leaf has the value zero.";
        }

        leaf hwm-rsp-time{
            type uint32;
            units "milliseconds";
            must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
                 " or " +
                 "(/pcep-state/entity/peers/peer/role = 'pcc'" +
                 " and hwm-rsp-time = 0))" {
                error-message
                    "Invalid greatest (high-water mark)
                     response time seen";
                description
                    "If role is pcc then this field is
                     meaningless and is set to zero.";
            }
            description
             "The greatest (high-water mark) response time seen.
              If no responses have been received then this object
              has the value zero.";
        }

        leaf num-pcreq-sent{
            type yang:counter32;
            description
             "The number of PCReq messages sent.";
        }
```

```
        leaf num-pcreq-rcvd{
            type yang:counter32;
            description
              "The number of PCReq messages received.";
        }

        leaf num-pcrep-sent{
            type yang:counter32;
            description
              "The number of PCRep messages sent.";
        }

        leaf num-pcrep-rcvd{
            type yang:counter32;
            description
              "The number of PCRep messages received.";
        }

        leaf num-pcerr-sent{
            type yang:counter32;
            description
              "The number of PCErr messages sent.";
        }

        leaf num-pcerr-rcvd{
            type yang:counter32;
            description
              "The number of PCErr messages received.";
        }

        leaf num-pcntf-sent{
            type yang:counter32;
            description
              "The number of PCNtf messages sent.";
        }

        leaf num-pcntf-rcvd{
            type yang:counter32;
            description
              "The number of PCNtf messages received.";
        }

        leaf num-keepalive-sent{
            type yang:counter32;
            description
              "The number of Keepalive messages sent.";
        }
```

```
        leaf num-keepalive-rcvd{
            type yang:counter32;
            description
              "The number of Keepalive messages received.";
        }

        leaf num-unknown-rcvd{
            type yang:counter32;
            description
              "The number of unknown messages received.";
        }

        leaf num-corrupt-rcvd{
            type yang:counter32;
            description
              "The number of corrupted PCEP message received.";
        }

        leaf num-req-sent{
            type yang:counter32;
            description
              "The number of requests sent.  A request corresponds
               1:1 with an RP object in a PCReq message. This might
               be greater than num-pcreq-sent because multiple
               requests can be batched into a single PCReq
               message.";
        }

        leaf num-req-sent-pend-rep{
            type yang:counter32;
            description
              "The number of requests that have been sent for
               which a response is still pending.";
        }

        leaf num-req-sent-ero-rcvd{
            type yang:counter32;
            description
               "The number of requests that have been sent for
                which a response with an ERO object was received.
                Such responses indicate that a path was
                successfully computed by the peer.";
        }

        leaf num-req-sent-nopath-rcvd{
            type yang:counter32;
            description
              "The number of requests that have been sent for
```

```
            which a response with a NO-PATH object was
            received. Such responses indicate that the peer
            could not  find a path to satisfy the
            request.";
        }

        leaf num-req-sent-cancel-rcvd{
            type yang:counter32;
            description
              "The number of requests that were cancelled with
               a PCNtf message.
               This might be different than num-pcntf-rcvd because
               not all PCNtf messages are used to cancel requests,
               and a single PCNtf message can cancel multiple
               requests.";
        }

        leaf num-req-sent-error-rcvd{
            type yang:counter32;
            description
             "The number of requests that were rejected with a
              PCErr message.
              This might be different than num-pcerr-rcvd because
              not all PCErr messages are used to reject requests,
              and a single PCErr message can reject multiple
              requests.";
        }

        leaf num-req-sent-timeout{
            type yang:counter32;
            description
             "The number of requests that have been sent to a peer
             and have been abandoned because the peer has taken too
             long to respond to them.";
        }

        leaf num-req-sent-cancel-sent{
            type yang:counter32;
            description
             "The number of requests that were sent to the peer and
              explicitly cancelled by the local PCEP entity sending
              a PCNtf.";
        }

        leaf num-req-rcvd{
            type yang:counter32;
            description
             "The number of requests received.  A request
```

```
              corresponds 1:1 with an RP object in a PCReq
              message.
              This might be greater than num-pcreq-rcvd because
              multiple requests can be batched into a single
              PCReq message.";
        }

        leaf num-req-rcvd-pend-rep{
            type yang:counter32;
            description
             "The number of requests that have been received for
              which a response is still pending.";
        }

        leaf num-req-rcvd-ero-sent{
            type yang:counter32;
            description
             "The number of requests that have been received for
              which a response with an ERO object was sent.  Such
              responses indicate that a path was successfully
              computed by the local PCEP entity.";
        }

        leaf num-req-rcvd-nopath-sent{
            type yang:counter32;
            description
             "The number of requests that have been received for
              which a response with a NO-PATH object was sent. Such
              responses indicate that the local PCEP entity could
              not find a path to satisfy the request.";
        }

        leaf num-req-rcvd-cancel-sent{
            type yang:counter32;
            description
             "The number of requests received that were cancelled
              by the local PCEP entity sending a PCNtf message.
              This might be different than num-pcntf-sent because
              not all PCNtf messages are used to cancel requests,
              and a single PCNtf message can cancel multiple
              requests.";
        }

        leaf num-req-rcvd-error-sent{
            type yang:counter32;
            description
             "The number of requests received that were cancelled
              by the local PCEP entity sending a PCErr message.
```

```
              This might be different than num-pcerr-sent because
              not all PCErr messages are used to cancel requests,
              and a single PCErr message can cancel multiple
              requests.";
        }

        leaf  num-req-rcvd-cancel-rcvd{
            type yang:counter32;
            description
             "The number of requests that were received from the
              peer and explicitly cancelled by the peer sending
              a PCNtf.";
        }

        leaf  num-rep-rcvd-unknown{
            type yang:counter32;
            description
              "The number of responses to unknown requests
               received. A response to an unknown request is a
               response whose RP object does not contain the
               request ID of any request that is currently
               outstanding on the session.";
        }

        leaf  num-req-rcvd-unknown{
            type yang:counter32;
            description
              "The number of unknown requests that have been
               received. An unknown request is a request
               whose RP object contains a request ID of
               zero.";
        }

        container svec{
            if-feature svec;
            description
                "If synchronized path computation is supported";
            leaf num-svec-sent{
                type yang:counter32;
                description
                  "The number of SVEC objects sent in PCReq messages.
                   An SVEC object represents a set of synchronized
                   requests.";
            }

            leaf num-svec-req-sent{
                type yang:counter32;
                description
```

```
                  "The number of requests sent that appeared in one
                   or more SVEC objects.";
            }

            leaf num-svec-rcvd{
                type yang:counter32;
                description
                 "The number of SVEC objects received in PCReq
                  messages. An SVEC object represents a set of
                  synchronized requests.";
            }

            leaf num-svec-req-rcvd{
                type yang:counter32;
                description
                  "The number of requests received that appeared
                   in one or more SVEC objects.";
            }
        }
        container stateful{
            if-feature stateful;
            description
                "Stateful PCE related statistics";
            leaf num-pcrpt-sent{
                type yang:counter32;
                description
                 "The number of PCRpt messages sent.";
            }

            leaf num-pcrpt-rcvd{
                type yang:counter32;
                description
                  "The number of PCRpt messages received.";
            }

            leaf num-pcupd-sent{
                type yang:counter32;
                description
                 "The number of PCUpd messages sent.";
            }

            leaf num-pcupd-rcvd{
                type yang:counter32;
                description
                  "The number of PCUpd messages received.";
            }

            leaf num-rpt-sent{
```

```
                type yang:counter32;
                description
                  "The number of LSP Reports sent.  A LSP report
                   corresponds 1:1 with an LSP object in a PCRpt
                   message. This might be greater than
                   num-pcrpt-sent because multiple reports can
                   be batched into a single PCRpt message.";
            }

            leaf num-rpt-rcvd{
                type yang:counter32;
                description
                 "The number of LSP Reports received.  A LSP report
                  corresponds 1:1 with an LSP object in a PCRpt
                  message.
                  This might be greater than num-pcrpt-rcvd because
                  multiple reports can be batched into a single
                  PCRpt message.";
            }

            leaf  num-rpt-rcvd-error-sent{
                type yang:counter32;
                description
                  "The number of reports of LSPs received that were
                  responded by the local PCEP entity by sending a
                  PCErr message.";
            }

            leaf num-upd-sent{
                type yang:counter32;
                description
                  "The number of LSP updates sent.  A LSP update
                   corresponds 1:1 with an LSP object in a PCUpd
                   message. This might be greater than
                   num-pcupd-sent because multiple updates can
                   be batched into a single PCUpd message.";
            }

            leaf num-upd-rcvd{
                type yang:counter32;
                description
                 "The number of LSP Updates received.  A LSP update
                  corresponds 1:1 with an LSP object in a PCUpd
                  message.
                  This might be greater than num-pcupd-rcvd because
                  multiple updates can be batched into a single
                  PCUpd message.";
            }
```

```
            leaf  num-upd-rcvd-unknown{
                type yang:counter32;
                description
                  "The number of updates to unknown LSPs
                   received. An update to an unknown LSP is a
                   update whose LSP object does not contain the
                   PLSP-ID of any LSP that is currently
                   present.";
            }

            leaf  num-upd-rcvd-undelegated{
                type yang:counter32;
                description
                  "The number of updates to not delegated LSPs
                   received. An update to an undelegated LSP is a
                   update whose LSP object does not contain the
                   PLSP-ID of any LSP that is currently
                   delegated to current PCEP session.";
            }

            leaf  num-upd-rcvd-error-sent{
                type yang:counter32;
                description
                  "The number of updates to LSPs received that were
                  responded by the local PCEP entity by sending a
                  PCErr message.";
            }
            container initiation {
                if-feature pce-initiated;
                description
                    "PCE-Initiated related statistics";
                leaf num-pcinitiate-sent{
                    type yang:counter32;
                    description
                     "The number of PCInitiate messages sent.";
                }

                leaf num-pcinitiate-rcvd{
                    type yang:counter32;
                    description
                      "The number of PCInitiate messages received.";
                }

                leaf num-initiate-sent{
                    type yang:counter32;
                    description
                      "The number of LSP Initiation sent via PCE.
                       A LSP initiation corresponds 1:1 with an LSP
```

```
                        object in a PCInitiate message. This might be
                        greater than num-pcinitiate-sent because
                        multiple initiations can be batched into a
                        single PCInitiate message.";
                }

                leaf num-initiate-rcvd{
                    type yang:counter32;
                    description
                      "The number of LSP Initiation received from
                       PCE.  A LSP initiation corresponds 1:1 with
                       an LSP object in a PCInitiate message. This
                       might be greater than num-pcinitiate-rcvd
                       because multiple initiations can be batched
                       into a single PCInitiate message.";
                }

                leaf  num-initiate-rcvd-error-sent{
                    type yang:counter32;
                    description
                      "The number of initiations of LSPs received
                       that were responded by the local PCEP entity
                       by sending a PCErr message.";
                }
            }
        }
    }//pcep-stats

    grouping lsp-state{
        description
            "This grouping defines the attributes for LSP in LSP-DB.
             These are the attributes specifically from the PCEP
             perspective";
        leaf plsp-id{
            type uint32{
                range "1..1048575";
            }
            description
                "A PCEP-specific identifier for the LSP.  A PCC
                 creates a unique PLSP-ID for each LSP that is
                 constant for the lifetime of a PCEP session.
                 PLSP-ID is 20 bits with 0 and 0xFFFFF are
                 reserved";
        }
        leaf pcc-id{
            type inet:ip-address;
            description
                "The local internet address of the PCC, that
```

```
                generated the PLSP-ID.";
        }

        container lsp-ref{
            description
                "reference to ietf-te lsp state";

            leaf source {
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:source";
                }
                description
                  "Tunnel sender address extracted from
                  SENDER_TEMPLATE  object";
                reference "RFC3209";
            }
            leaf destination {
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:"
                         + "destination";
                }
                description
                    "Tunnel endpoint address extracted from
                    SESSION object";
                reference "RFC3209";
            }
            leaf tunnel-id {
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:tunnel-id";
                }
                description
                    "Tunnel identifier used in the SESSION
                    that remains constant over the life
                    of the tunnel.";
                reference "RFC3209";
            }
            leaf lsp-id {
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:lsp-id";
                }
                description
                    "Identifier used in the SENDER_TEMPLATE
                    and the FILTER_SPEC that can be changed
                    to allow a sender to share resources with
                    itself.";
                reference "RFC3209";
            }
            leaf extended-tunnel-id {
```

```
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:"
                    + "extended-tunnel-id";
                }
                description
                    "Extended Tunnel ID of the LSP.";
                reference "RFC3209";
            }
            leaf type {
                type leafref {
                    path "/te:te/te:lsps-state/te:lsp/te:type";
                }
                description "LSP type P2P or P2MP";
            }
        }

        leaf admin-state{
            type boolean;
            description
                "The desired operational state";
        }
        leaf operational-state{
            type operational-state;
            description
                "The operational status of the LSP";
        }
        container delegated{
            description
                "The delegation related parameters";
            leaf enabled{
                type boolean;
                description
                    "LSP is delegated or not";
            }
            leaf pce{
                type leafref {
                    path "/pcep-state/entity/peers/peer/addr";
                }
                must "((../enabled == true)" +
                     " and " +
                     "((../../role == 'pcc')" +
                     " or " +
                    "(../../role == 'pcc-and-pce')))"
                {
                    error-message
                        "The PCEP entity must be PCC
                         and the LSP be delegated";
                    description
```

```
                    "When PCEP entity is PCC for
                    delegated LSP";
                }
                description
                    "The reference to the PCE peer to
                    which LSP is delegated";
            }
            leaf srp-id{
                type uint32;
                description
                    "The last SRP-ID-number associated with this
                    LSP.";
            }
        }
        container initiation {
            if-feature pce-initiated;
            description
                "The PCE initiation related parameters";
            leaf enabled{
                type boolean;
                description
                    "LSP is PCE-initiated or not";
            }
            leaf pce{
                type leafref {
                    path "/pcep-state/entity/peers/peer/addr";
                }
                must "(../enabled == true)"
                {
                    error-message
                        "The LSP must be PCE-Initiated";
                    description
                        "When the LSP must be PCE-Initiated";
                }
                description
                    "The reference to the PCE
                    that initiated this LSP";
            }
        }
        leaf symbolic-path-name{
            type string;
            description
                "The symbolic path name associated with the LSP.";
        }
        leaf last-error{
            type lsp-error;
            description
                "The last error for the LSP.";
```

```
        }
                leaf pst{
                        type pst;
                        default "rsvp-te";
                        description
                                "The Path Setup Type";

                }

    }//lsp-state

    grouping notification-instance-hdr {
        description
            "This group describes common instance specific data
             for notifications.";

        leaf peer-addr {
            type leafref {
                path "/pcep-state/entity/peers/peer/addr";
            }
            description
                "Reference to peer address";
        }

    }// notification-instance-hdr

    grouping notification-session-hdr {
        description
        "This group describes common session instance specific
         data for notifications.";

        leaf session-initiator {
            type leafref {
            path "/pcep-state/entity/peers/peer/sessions/" +
                 "session/initiator";
            }
            description
                "Reference to pcep session initiator leaf";
        }
    }// notification-session-hdr

    grouping stateful-pce-parameter {
        description
        "This group describes stateful PCE specific
         parameters.";
        leaf state-timeout{
            type uint32;
            units "seconds";
```

```
            description
                "When a PCEP session is terminated, a PCC
                 waits for this time period before flushing
                 LSP state associated with that PCEP session
                 and reverting to operator-defined default
                 parameters or behaviours.";
        }
        leaf redelegation-timeout{
            type uint32;
            units "seconds";
            must "((../role == 'pcc')" +
                 " or " +
                 "(../role == 'pcc-and-pce'))"
            {
                error-message "The PCEP entity must be PCC";
                description
                    "When PCEP entity is PCC";
            }
            description
                "When a PCEP session is terminated, a PCC
                 waits for this time period before revoking
                 LSP delegation to a PCE and attempting to
                 redelegate LSPs associated with the
                 terminated PCEP session to an alternate
                 PCE.";
        }
        leaf rpt-non-pcep-lsp{
            type boolean;
            must "((../role == 'pcc')" +
                 " or " +
                 "(../role == 'pcc-and-pce'))"
            {
                error-message "The PCEP entity must be PCC";
                description
                    "When PCEP entity is PCC";
            }
            description
                "If set, a PCC reports LSPs that are not
                controlled by any PCE (for example, LSPs
                that are statically configured at the
                PCC). ";
        }

    }

    grouping authentication {
        description "Authentication Information";
        choice auth-type-selection {
```

```
            description
                "Options for expressing authentication setting.";
            case auth-key-chain {
                leaf key-chain {
                    type key-chain:key-chain-ref;
                    description
                        "key-chain name.";
                }
            }
            case auth-key {
                leaf key {
                    type string;
                description
                    "Key string in ASCII format.";
                }
                container crypto-algorithm {
                    uses key-chain:crypto-algorithm-types;
                        description
                            "Cryptographic algorithm associated
                             with key.";
                }
            }
            case auth-tls {
                if-feature tls;
                container tls {
                    description
                        "TLS related information - TBD";
                }
            }
        }
    }

    grouping association {
        description
            "Generic Association parameters";
        leaf type {
            type "assoc-type";
            description
                "The PCEP association type";
        }
        leaf id {
            type uint16;
            description
                "PCEP Association ID";
        }
        leaf source {
          type inet:ip-address;
          description
```

```
                "PCEP Association Source.";
        }
        leaf global-source {
          type uint32;
          description
                "PCEP Association Global
                Source.";
        }
        leaf extended-id{
            type string;
            description
                "Additional information to
                support unique identification.";
        }
    }
    grouping association-ref {
        description
            "Generic Association parameters";
        leaf id {
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                + "association-list/id";
            }
            description
                "PCEP Association ID";
        }
        leaf source {
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                + "association-list/source";
            }
          description
                "PCEP Association Source.";
        }
        leaf global-source {
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                + "association-list/global-source";
            }
          description
                "PCEP Association Global
                Source.";
        }
        leaf extended-id{
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                + "association-list/extended-id";
            }
```

```
            description
                "Additional information to
                support unique identification.";
        }
    }
    /*
     * Configuration data nodes
     */
    container pcep{

        presence
            "The PCEP is enabled";

            description
            "Parameters for list of configured PCEP entities
             on the device.";

        container entity {

            description
                "The configured PCEP entity on the device.";

            leaf addr {
                type inet:ip-address;
                mandatory true;
                description
                    "The local Internet address of this PCEP
                    entity.
                    If operating as a PCE server, the PCEP
                    entity listens on this address.
                    If operating as a PCC, the PCEP entity
                    binds outgoing TCP connections to this
                    address.
                    It is possible for the PCEP entity to
                    operate both as a PCC and a PCE Server, in
                    which case it uses this address both to
                    listen for incoming TCP connections and to
                    bind outgoing TCP connections.";
            }

            leaf enabled {
                type boolean;
                default true;
                description
                    "The administrative status of this PCEP
                     Entity.";
            }
```

```
            leaf role {
                type pcep-role;
                mandatory true;
                description
                    "The role that this entity can play.
                     Takes one of the following values.
                     - unknown(0): this PCEP Entity role is not
                       known.
                     - pcc(1): this PCEP Entity is a PCC.
                     - pce(2): this PCEP Entity is a PCE.
                     - pcc-and-pce(3): this PCEP Entity is both
                       a PCC and a PCE.";
            }

            leaf description {
                type string;
                description
                    "Description of the PCEP entity configured
                     by the user";
            }

            uses info {
                description
                    "Local PCEP entity information";
            }


            container pce-info {
                must "((../role == 'pce')" +
                     " or " +
                     "(../role == 'pcc-and-pce'))"
                {
                    error-message "The PCEP entity must be PCE";
                    description
                        "When PCEP entity is PCE";
                }
                uses pce-info {
                    description
                        "Local PCE information";
                }
                                uses authentication {
                                        description
                                                "Local PCE authentication inform
ation";
                                }


                description
                    "The Local PCE Entity PCE information";
```

```
            }

            uses pcep-entity-info {
                description
                    "The configuration related to the PCEP
                     entity.";
            }



            leaf pcep-notification-max-rate {
                type uint32;
                mandatory true;
                description
                    "This variable indicates the maximum number of
                     notifications issued per second. If events occur
                     more rapidly, the implementation may simply fail
                     to emit these notifications during that period,
                     or may queue them until an appropriate time. A
                     value of 0 means no notifications are emitted
                     and all should be discarded (that is, not
                     queued).";
            }

            container stateful-parameter{
                if-feature stateful;
                must "(../info/capability/stateful/active == true)"
                {
                    error-message
                        "The Active Stateful PCE must be enabled";
                    description
                        "When PCEP entity is active stateful
                         enabled";
                }
                uses stateful-pce-parameter;

                description
                    "The configured stateful parameters";
            }


            container peers{
                must "((../role == 'pcc')" +
                     " or " +
                     "(../role == 'pcc-and-pce'))"
                {
                    error-message
                        "The PCEP entity must be PCC";
```

```
                    description
                        "When PCEP entity is PCC, as remote
                         PCE peers are configured.";
                }
                description
                    "The list of configured peers for the
                     entity (remote PCE)";
                list peer{
                    key "addr";

                    description
                        "The peer configured for the entity.
                         (remote PCE)";

                    leaf addr {
                        type inet:ip-address;
                        description
                            "The local Internet address of this
                             PCEP peer.";
                    }

                    leaf description {
                        type string;
                        description
                            "Description of the PCEP peer
                             configured by the user";
                    }
                    uses info {
                        description
                            "PCE Peer information";
                    }
                    uses pce-info {
                        description
                            "PCE Peer information";
                    }

                    leaf delegation-pref{
                        if-feature stateful;
                        type uint8{
                            range "0..7";
                        }
                        must "(../../info/capability/stateful/active"
                             + "== true)"
                        {
                            error-message
                                "The Active Stateful PCE must be
                                 enabled";
                            description
```

```
                                "When PCEP entity is active stateful
                                 enabled";
                            }
                            description
                                "The PCE peer delegation preference.";
                        }
                        uses authentication {
                            description
                                "PCE Peer authentication";
                        }
                    }//peer
                }//peers
            }//entity
        }//pcep

        /*
         * Operational data nodes
         */

        container pcep-state{
            config false;
            description
                "The list of operational PCEP entities on the
                 device.";

            container entity{
                description
                    "The operational PCEP entity on the device.";

                leaf addr {
                    type inet:ip-address;
                    description
                        "The local Internet address of this PCEP
                        entity.
                        If operating as a PCE server, the PCEP
                        entity listens on this address.
                        If operating as a PCC, the PCEP entity
                        binds outgoing TCP connections to this
                        address.
                        It is possible for the PCEP entity to
                        operate both as a PCC and a PCE Server, in
                        which case it uses this address both to
                        listen for incoming TCP connections and to
                        bind outgoing TCP connections.";
                }

                leaf index{
                    type uint32;
```

```
            description
                "The index of the operational PECP
                 entity";
        }


        leaf admin-status {
            type pcep-admin-status;
            description
                "The administrative status of this PCEP Entity.
                 This is the desired operational status as
                 currently set by an operator or by default in
                 the implementation.  The value of enabled
                 represents the current status of an attempt
                 to reach this desired status.";
        }

        leaf oper-status {
            type pcep-admin-status;
            description
               "The operational status of the PCEP entity.
                Takes one of the following values.
                - oper-status-up(1): the PCEP entity is
                  active.
                - oper-status-down(2): the PCEP entity is
                  inactive.
                - oper-status-going-up(3): the PCEP entity is
                  activating.
                - oper-status-going-down(4): the PCEP entity is
                  deactivating.
                - oper-status-failed(5): the PCEP entity has
                  failed and will recover when possible.
                - oper-status-failed-perm(6): the PCEP entity
                  has failed and will not recover without
                  operator intervention.";
        }

        leaf role {
            type pcep-role;
            description
                "The role that this entity can play.
                 Takes one of the following values.
                 - unknown(0): this PCEP entity role is
                   not known.
                 - pcc(1): this PCEP entity is a PCC.
                 - pce(2): this PCEP entity is a PCE.
                 - pcc-and-pce(3): this PCEP entity is
                   both a PCC and a PCE.";
```

```
            }

            uses info {
                description
                    "Local PCEP entity information";
            }

            container pce-info {
                when "((../role == 'pce')" +
                     " or " +
                     "(../role == 'pcc-and-pce'))"
                {
                    description
                        "When PCEP entity is PCE";
                }
                uses pce-info {
                    description
                        "Local PCE information";
                }
                                uses authentication {
                                        description
                                                "Local PCE authentication inform
ation";
                                }
                description
                    "The Local PCE Entity PCE information";
            }

            uses pcep-entity-info{
                description
                    "The operational information related to the
                     PCEP entity.";
            }

            container stateful-parameter{
                if-feature stateful;
                must "(../info/capability/stateful/active == true)"
                {
                    error-message
                        "The Active Stateful PCE must be enabled";
                    description
                        "When PCEP entity is active stateful
                         enabled";
                }
                uses stateful-pce-parameter;

                description
                    "The operational stateful parameters";
            }
```

```
            container lsp-db{
                if-feature stateful;
                description
                    "The LSP-DB";
                list association-list {
                    key "id source global-source extended-id";
                    description
                        "List of all PCEP associations";
                    uses association {
                        description
                            "The Association attributes";
                    }
                    list lsp {
                        key "plsp-id pcc-id";
                        description
                            "List of all LSP in this association";
                        leaf plsp-id {
                            type leafref {
                                path "/pcep-state/entity/lsp-db/"
                                + "lsp/plsp-id";
                            }
                            description
                                "Reference to PLSP-ID in LSP-DB";
                        }
                        leaf pcc-id {
                            type leafref {
                                path "/pcep-state/entity/lsp-db/"
                                + "lsp/pcc-id";
                            }
                            description
                                "Reference to PCC-ID in LSP-DB";
                        }
                    }
                }
                list lsp{
                    key "plsp-id pcc-id";
                    description
                        "List of all LSPs in LSP-DB";
                    uses lsp-state{
                        description
                            "The PCEP specific attributes for
                             LSP-DB.";
                    }
                    list association-list {
                        key "id source global-source extended-id";
                        description
                            "List of all PCEP associations";
                        uses association-ref {
```

```
                        description
                            "Reference to the Association
                             attributes";
                    }
                }

            }
        }
        container peers{
            description
                    "The list of peers for the entity";

            list peer{
                key "addr";

                description
                    "The peer for the entity.";

                leaf addr {
                    type inet:ip-address;
                    description
                        "The local Internet address of this PCEP
                         peer.";
                }

                leaf role {
                    type pcep-role;
                    description
                        "The role of the PCEP Peer.
                         Takes one of the following values.
                         - unknown(0): this PCEP peer role
                           is not known.
                         - pcc(1): this PCEP peer is a PCC.
                         - pce(2): this PCEP peer is a PCE.
                         - pcc-and-pce(3): this PCEP peer
                           is both a PCC and a PCE.";
                }


                uses info {
                    description
                        "PCEP peer information";
                }


                container pce-info {
                    when "((../role == 'pce')" +
                    " or " +
```

```
            "(../role == 'pcc-and-pce'))"
            {
                description
                    "When PCEP entity is PCE";
            }
            uses pce-info {
                description
                    "PCE Peer information";
            }
        description
            "The PCE Peer information";
        }

        leaf delegation-pref{
            if-feature stateful;
            type uint8{
                range "0..7";
            }
            must "((../../role == 'pcc')" +
                 " or " +
                 "(../../role == 'pcc-and-pce'))"
            {
                error-message
                    "The PCEP entity must be PCC";
                description
                    "When PCEP entity is PCC";
            }
            must "(../../info/capability/stateful/active"
                 + " == true)"
            {
                error-message
                    "The Active Stateful PCE must be
                     enabled";
                description
                    "When PCEP entity is active stateful
                     enabled";
            }
            description
                "The PCE peer delegation preference.";
        }

        uses authentication {
            description
                "PCE Peer authentication";
        }

        leaf discontinuity-time {
            type yang:timestamp;
```

```
            description
                "The timestamp of the time when the
                 information and statistics were
                 last reset.";
        }

        leaf initiate-session {
            type boolean;
            description
                "Indicates whether the local PCEP
                 entity initiates sessions to this peer,
                 or waits for the peer to initiate a
                 session.";
        }

        leaf session-exists{
            type boolean;
            description
                "Indicates whether a session with
                 this peer currently exists.";
        }

        leaf num-sess-setup-ok{
            type yang:counter32;
            description
                "The number of PCEP sessions successfully
                 successfully established with the peer,
                 including any current session.  This
                 counter is incremented each time a
                 session with this peer is successfully
                 established.";
        }

        leaf num-sess-setup-fail{
            type yang:counter32;
            description
               "The number of PCEP sessions with the peer
                that have been attempted but failed
                before being fully established. This
                counter is incremented each time a
                session retry to this peer fails.";
        }

        leaf session-up-time{
            type yang:timestamp;
            must "(../num-sess-setup-ok != 0  or " +
                "(../num-sess-setup-ok = 0  and "  +
                "session-up-time = 0))" {
```

```
                    error-message
                        "Invalid Session Up timestamp";
                    description
                        "If num-sess-setup-ok is zero,
                         then this leaf contains zero.";
                }
            description
               "The timestamp value of the last time a
                session with this peer was successfully
                established.";
        }

        leaf session-fail-time{
            type yang:timestamp;
            must "(../num-sess-setup-fail != 0  or " +
                "(../num-sess-setup-fail = 0  and "  +
                "session-fail-time = 0))" {
                    error-message
                        "Invalid Session Fail timestamp";
                    description
                        "If num-sess-setup-fail is zero,
                         then this leaf contains zero.";
                }
            description
              "The timestamp value of the last time a
               session with this peer failed to be
               established.";
        }

        leaf session-fail-up-time{
            type yang:timestamp;
            must "(../num-sess-setup-ok != 0  or "     +
                "(../num-sess-setup-ok = 0  and "     +
                "session-fail-up-time = 0))" {
                    error-message
                        "Invalid Session Fail from
                         Up timestamp";
                    description
                        "If num-sess-setup-ok is zero,
                         then this leaf contains zero.";
                }
            description
              "The timestamp value of the last time a
               session with this peer failed from
               active.";
        }

        container pcep-stats {
```

```
                description
                    "The container for all statistics at peer
                     level.";
                uses pcep-stats{
                    description
                        "Since PCEP sessions can be
                        ephemeral, the peer statistics tracks
                        a peer even when no PCEP session
                        currently exists to that peer. The
                        statistics contained are an aggregate
                        of the statistics for all successive
                        sessions to that peer.";
                }

                leaf num-req-sent-closed{
                    type yang:counter32;
                    description
                        "The number of requests that were
                         sent to the peer and implicitly
                         cancelled when the session they were
                         sent over was closed.";
                }

                leaf  num-req-rcvd-closed{
                    type yang:counter32;
                    description
                        "The number of requests that were
                         received from the peer and
                         implicitly cancelled when the
                         session they were received over
                         was closed.";
                }
            }//pcep-stats



            container sessions {
                description
                    "This entry represents a single PCEP
                     session in which the local PCEP entity
                     participates.
                     This entry exists only if the
                     corresponding PCEP session has been
                     initialized by some event, such as
                     manual user configuration, auto-
                     discovery of a peer, or an incoming
                     TCP connection.";
```

```
                    list session {
                        key "initiator";

                        description
                            "The list of sessions, note that
                             for a time being two sessions
                             may exist for a peer";

                        leaf initiator {
                            type pcep-initiator;
                            description
                                "The initiator of the session,
                                 that is, whether the TCP
                                 connection was initiated by
                                 the local PCEP entity or the
                                 peer.
                                 There is a window during
                                 session initialization where
                                 two sessions can exist between
                                 a pair of PCEP speakers, each
                                 initiated by one of the
                                 speakers. One of these
                                 sessions is always discarded
                                 before it leaves OpenWait state.
                                 However, before it is discarded,
                                 two sessions to the given peer
                                 appear transiently in this MIB
                                 module. The sessions are
                                 distinguished by who initiated
                                 them, and so this field is the
                                 key.";
                        }

                        leaf state-last-change {
                            type yang:timestamp;
                            description
                                "The timestamp value at the
                                 time this session entered its
                                 current state as denoted by
                                 the state leaf.";
                        }

                        leaf state {
                            type pcep-sess-state;
                            description
                                "The current state of the
                                 session.
                                 The set of possible states
```

```
                excludes the idle state since
                entries do not exist in the
                idle state.";
            }

            leaf session-creation {
                type yang:timestamp;
                description
                    "The timestamp value at the
                     time this session was
                     created.";
            }

            leaf connect-retry {
                type yang:counter32;
                description
                     "The number of times that the
                      local PCEP entity has
                      attempted to establish a TCP
                      connection for this session
                      without success. The PCEP
                      entity gives up when this
                      reaches connect-max-retry.";
            }

            leaf local-id {
                type uint32 {
                  range "0..255";
                }
                description
                     "The value of the PCEP session
                      ID used by the local PCEP
                      entity in the Open message
                      for this session.
                      If state is tcp-pending then
                      this is the session ID that
                      will be used in the Open
                      message. Otherwise, this is
                      the session ID that was sent
                      in the Open message.";
            }

            leaf remote-id {
                type uint32 {
                    range "0..255";
                }
                must "((../state != 'tcp-pending'" +
                     "and " +
```

```
                    "../state != 'open-wait' )" +
                    "or " +
                    "((../state = 'tcp-pending'" +
                    " or " +
                    "../state = 'open-wait' )" +
                    "and remote-id = 0))" {
                       error-message
                           "Invalid remote-id";
                       description
                            "If state is tcp-pending
                             or open-wait then this
                             leaf is not used and
                             MUST be set to zero.";
                    }
                description
                    "The value of the PCEP session
                     ID used by the peer in its
                     Open message for this
                     session.";
            }

            leaf keepalive-timer {
                type uint32 {
                  range "0..255";
                }
                units "seconds";
                must "(../state = 'session-up'" +
                    "or " +
                    "(../state != 'session-up'" +
                    "and keepalive-timer = 0))" {
                       error-message
                           "Invalid keepalive
                            timer";
                       description
                           "This field is used if
                            and only if state is
                            session-up. Otherwise,
                            it is not used and
                            MUST be set to
                            zero.";
                    }
                description
                    "The agreed maximum interval at
                     which the local PCEP entity
                     transmits PCEP messages on this
                     PCEP session.  Zero means that
                     the local PCEP entity never
                     sends Keepalives on this
```

```
                session.";
            }

            leaf peer-keepalive-timer {
                type uint32 {
                  range "0..255";
                }
                units "seconds";
                must "(../state = 'session-up'" +
                     "or "   +
                     "(../state != 'session-up'" +
                     "and " +
                     "peer-keepalive-timer = 0))" {
                        error-message
                            "Invalid Peer keepalive
                             timer";
                        description
                            "This field is used if
                             and only if state is
                             session-up. Otherwise,
                             it is not used and MUST
                             be set to zero.";
                    }
                description
                     "The agreed maximum interval at
                      which the peer transmits PCEP
                      messages on this PCEP session.
                      Zero means that the peer never
                      sends Keepalives on this
                      session.";
            }

            leaf dead-timer {
                type uint32 {
                  range "0..255";
                }
                units "seconds";
                description
                     "The dead timer interval for
                      this PCEP session.";
            }

            leaf peer-dead-timer {
                type uint32 {
                  range "0..255";
                }
                units "seconds";
                must "((../state != 'tcp-pending'" +
```

```
                    "and " +
                    "../state != 'open-wait' )" +
                    "or " +
                    "((../state = 'tcp-pending'" +
                    " or " +
                    "../state = 'open-wait' )" +
                    "and " +
                    "peer-dead-timer = 0))" {
                       error-message
                           "Invalid Peer Dead
                            timer";
                       description
                           "If state is tcp-
                            pending or open-wait
                            then this leaf is not
                            used and MUST be set to
                            zero.";
                    }
                description
                    "The peer's dead-timer interval
                     for this PCEP session.";
            }

            leaf ka-hold-time-rem {
                type uint32 {
                  range "0..255";
                }
                units "seconds";
                must "((../state != 'tcp-pending'" +
                    "and " +
                    "../state != 'open-wait' ) " +
                    "or " +
                    "((../state = 'tcp-pending'" +
                    "or " +
                    "../state = 'open-wait' )" +
                    "and " +
                    "ka-hold-time-rem = 0))" {
                       error-message
                           "Invalid Keepalive hold
                            time remaining";
                       description
                           "If state is tcp-pending
                            or open-wait then this
                            field is not used and
                            MUST be set to zero.";
                    }
                description
                    "The keep alive hold time
```

```
                remaining for this session.";
            }

            leaf overloaded {
                type boolean;
                description
                     "If the local PCEP entity has
                      informed the peer that it is
                      currently overloaded, then this
                      is set to true.  Otherwise, it
                      is set to false.";
            }

            leaf overload-time {
                type uint32;
                units "seconds";
                must "(../overloaded = true or" +
                    "(../overloaded != true and" +
                    " overload-time = 0))" {
                      error-message
                          "Invalid overload-time";
                      description
                          "This field is only used
                           if overloaded is set to
                           true. Otherwise, it is
                           not used and MUST be set
                           to zero.";
                    }
                description
                     "The interval of time that is
                      remaining until the local PCEP
                      entity will cease to be
                      overloaded on this session.";
            }

            leaf peer-overloaded {
                type boolean;
                description
                     "If the peer has informed the
                      local PCEP entity that it is
                      currently overloaded, then this
                      is set to true. Otherwise, it
                      is set to false.";
            }

            leaf peer-overload-time {
                type uint32;
                units "seconds";
```

```
            must "(../peer-overloaded = true" +
                 " or " +
                 "(../peer-overloaded != true" +
                 " and " +
                 "peer-overload-time = 0))" {
                    error-message
                        "Invalid peer overload
                         time";
                    description
                        "This field is only used
                         if peer-overloaded is
                         set to true. Otherwise,
                         it is not used and MUST
                         be set to zero.";
                }
            description
                 "The interval of time that is
                  remaining until the peer will
                  cease to be overloaded.  If it
                  is not known how long the peer
                  will stay in overloaded state,
                  this leaf is set to zero.";
        }
        leaf lspdb-sync {
            if-feature stateful;
            type sync-state;
            description
                "The LSP-DB state synchronization
                status.";
        }
        leaf discontinuity-time {
            type yang:timestamp;
            description
                 "The timestamp value of the time
                  when the statistics were last
                  reset.";
        }

        container pcep-stats {
            description
                "The container for all statistics
                 at session level.";
            uses pcep-stats{
                description
                    "The statistics contained are
                     for the current sessions to
                     that peer. These are lost
                     when the session goes down.
```

```
                                        ";
                                    }
                                }//pcep-stats

                            } // session
                        } // sessions
                    }//peer
                }//peers
            }//entity
        }//pcep-state

        /*
         * Notifications
         */
        notification pcep-session-up {
            description
                "This notification is sent when the value of
                 '/pcep/pcep-state/peers/peer/sessions/session/state'
                 enters the 'session-up' state.";

            uses notification-instance-hdr;

            uses notification-session-hdr;

            leaf state-last-change {
                type yang:timestamp;
                description
                    "The timestamp value at the time this session entered
                    its current state as denoted by the state leaf.";
            }

            leaf state {
                type pcep-sess-state;
                description
                    "The current state of the session.
                     The set of possible states excludes the idle state
                     since entries do not exist in the idle state.";
            }
        } //notification

        notification pcep-session-down {
            description
                "This notification is sent when the value of
                 '/pcep/pcep-state/peers/peer/sessions/session/state'
                 leaves the 'session-up' state.";

            uses notification-instance-hdr;
```

```
        leaf session-initiator {
            type pcep-initiator;
            description
                "The initiator of the session.";
        }

        leaf state-last-change {
            type yang:timestamp;
            description
                "The timestamp value at the time this session entered
                its current state as denoted by the state leaf.";
        }

        leaf state {
            type pcep-sess-state;
            description
                "The current state of the session.
                 The set of possible states excludes the idle state
                 since entries do not exist in the idle state.";
        }
    } //notification

    notification pcep-session-local-overload {
        description
            "This notification is sent when the local PCEP entity
            enters overload state for a peer.";

        uses notification-instance-hdr;

        uses notification-session-hdr;

        leaf overloaded {
            type boolean;
            description
                 "If the local PCEP entity has informed the peer that
                 it is currently overloaded, then this is set to
                 true. Otherwise, it is set to false.";
        }

        leaf overload-time {
            type uint32;
            units "seconds";
            must "(../overloaded = true or "    +
                "(../overloaded != true  and "  +
                "overload-time = 0))" {
                    error-message
                        "Invalid overload-time";
                    description
```

```
                        "This field is only used if overloaded is
                         set to true. Otherwise, it is not used
                         and MUST be set to zero.";
                    }
            description
                 "The interval of time that is remaining until the
                 local PCEP entity will cease to be overloaded on
                 this session.";
        }
    } //notification

    notification pcep-session-local-overload-clear {
        description
            "This notification is sent when the local PCEP entity
            leaves overload state for a peer.";

        uses notification-instance-hdr;

        leaf overloaded {
            type boolean;
            description
                 "If the local PCEP entity has informed the peer
                 that it is currently overloaded, then this is set
                 to true.  Otherwise, it is set to false.";
        }
    } //notification

    notification pcep-session-peer-overload {
        description
            "This notification is sent when a peer enters overload
            state.";

        uses notification-instance-hdr;

        uses notification-session-hdr;

        leaf peer-overloaded {
            type boolean;
            description
                "If the peer has informed the local PCEP entity that
                it is currently overloaded, then this is set to true.
                Otherwise, it is set to false.";
        }

        leaf peer-overload-time {
            type uint32;
            units "seconds";
            must "(../peer-overloaded = true or "    +
```

```
                "(../peer-overloaded != true  and "  +
                "peer-overload-time = 0))" {
                    error-message
                        "Invalid peer-overload-time";
                    description
                        "This field is only used if
                         peer-overloaded is set to true.
                         Otherwise, it is not used and MUST
                         be set to zero.";
                }
            description
                "The interval of time that is remaining until the
                peer will cease to be overloaded.  If it is not known
                how long the peer will stay in overloaded state, this
                leaf is set to zero.";
        }
    } //notification

    notification pcep-session-peer-overload-clear {
        description
            "This notification is sent when a peer leaves overload
            state.";

        uses notification-instance-hdr;

        leaf peer-overloaded {
            type boolean;
            description
                "If the peer has informed the local PCEP entity that
                it is currently overloaded, then this is set to true.
                Otherwise, it is set to false.";
        }
    } //notification
}//module

<CODE ENDS>
```

9. 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 the 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.

TBD: List specific Subtrees and data nodes and their sensitivity/ vulnerability.

10. Manageability Considerations

10.1. Control of Function and Policy

10.2. Information and Data Models

10.3. Liveness Detection and Monitoring

10.4. Verify Correct Operations

10.5. Requirements On Other Protocols

10.6. Impact On Network Operations

11. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

URI: urn:ietf:params:xml:ns:yang:ietf-pcep

Registrant Contact: The PCE WG of the IETF.

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-pcep
Namespace:    urn:ietf:params:xml:ns:yang:ietf-pcep
Prefix:       pcep
Reference:    This I-D
```

12. Acknowledgements

The initial document is based on the PCEP MIB [RFC7420]. Further this document structure is based on Routing Yang Module [I-D.ietf-netmod-routing-cfg]. We would like to thank the authors of aforementioned documents.

13. References

13.1. 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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <http://www.rfc-editor.org/info/rfc5440>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <http://www.rfc-editor.org/info/rfc6991>.

[I-D.ietf-pce-stateful-pce] Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-14 (work in progress), March 2016.

[I-D.ietf-pce-pce-initiated-lsp] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-ietf-pce-pce-initiated-lsp-05 (work in progress), October 2015.

[I-D.ietf-pce-lsp-setup-type]
Sivabalan, S., Medved, J., Minei, I., Crabbe, E., Varga, R., Tantsura, J., and J. Hardwick, "Conveying path setup type in PCEP messages", draft-ietf-pce-lsp-setup-type-03 (work in progress), June 2015.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., Lopez, V., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-07 (work in progress), March 2016.

13.2. Informative References

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <http://www.rfc-editor.org/info/rfc4655>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

[RFC7420] Koushik, A., Stephan, E., Zhao, Q., King, D., and J. Hardwick, "Path Computation Element Communication Protocol (PCEP) Management Information Base (MIB) Module", RFC 7420, DOI 10.17487/RFC7420, December 2014, <http://www.rfc-editor.org/info/rfc7420>.

[I-D.ietf-netmod-routing-cfg]
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-22 (work in progress), July 2016.

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-06 (work in progress), March 2016.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-03 (work in progress), March 2016.

Appendix A. Contributor Addresses

Rohit Pobbathi
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: rohit.pobbathi@huawei.com

Vinod KumarS
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: vinods.kumar@huawei.com

Zafar Ali
Cisco Systems
Canada

EMail: zali@cisco.com

Xufeng Liu
Ericsson
1595 Spring Hill Road, Suite 500
Vienna, VA 22182
USA

EMail: xufeng.liu@ericsson.com

Young Lee
Huawei Technologies
5340 Legacy Drive, Building 3
Plano, TX 75023, USA

Phone: (469) 277-5838
EMail: leeyoung@huawei.com

Udayasree Palle
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: udayasree.palle@huawei.com

Xian Zhang
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R.China

EMail: zhang.xian@huawei.com

Avantika
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: avantika.sushilkumar@huawei.com

Authors' Addresses

Dhruv Dhody (editor)
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: dhruv.ietf@gmail.com

Jonathan Hardwick
Metaswitch
100 Church Street
Enfield EN2 6BQ
UK

EMail: jonathan.hardwick@metaswitch.com

Vishnu Pavan Beeram
Juniper Networks
USA

EMail: vbeeram@juniper.net

Jeff Tantsura
USA

EMail: jefftant@gmail.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2017

K. Raza
R. Asati
Cisco Systems, Inc.

X. Liu
Ericsson

S. Esale
Juniper Networks

X. Chen
Huawei Technologies

H. Shah
Ciena Corporation

July 8, 2016

YANG Data Model for MPLS LDP and mLDP
draft-raza-mpls-ldp-mldp-yang-04

Abstract

This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on January 9, 2017.

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modelling language for the NETCONF.

This document introduces a YANG data model for MPLS Label Distribution Protocol (LDP) [RFC5036] and Multipoint LDP (mLDP) [RFC6388]. For LDP, it also covers LDP IPv6 [RFC7552] and LDP capabilities [RFC5561].

The data model is defined for following constructs that are used for managing the protocol:

- Configuration
- Operational State
- Executables (Actions)
- Notifications

This document is organized to define the data model for each of the above constructs (configuration, state, action, and notifications) in the sequence as listed earlier. Given that mLDP is tightly coupled with LDP, mLDP data model is defined under LDP tree and in the same sequence as listed above.

2. Specification of Requirements

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

In this document, the word "IP" is used to refer to both IPv4 and IPv6, unless otherwise explicitly stated. For example, "IP address family" means and be read as "IPv4 and/or IPv6 address family"

3. LDP YANG Model

3.1. Overview

This document defines a new module named "ietf-mpls-ldp" for LDP/mLDP data model where this module augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg].

There are four main containers in "ietf-mpls-ldp" module as follows:

- Read-Write parameters for configuration (Discussed in Section 3.2)
- Read-only parameters for operational state (Discussed in Section 3.3)
- Notifications for events (Discussed in Section 3.4)
- RPCs for executing commands to perform some action (Discussed in Section 3.5)

For the configuration and state data, this model follows the similar approach described in [I-D.openconfig-netmod-opstate] to represent the configuration (intended state) and operational (applied and derived) state. This means that for every configuration (rw) item, there is an associated (ro) item under "state" container to represent the applied state. Furthermore, protocol derived state is also kept under "state" tree corresponding to the protocol area (discovery, peer etc.). [Ed note: This document will be (re-)aligned with [I-D.openconfig-netmod-opstate] once that specification is adopted as a WG document]

Following diagram depicts high level LDP yang tree organization and hierarchy:

```
module: ietf-mpls-ldp
     +-- rw routing
       +-- rw control-plane-protocols
          +-- rw mpls-ldp
              +-- rw global
              |   +-- rw config
              |       +-- rw ...
              |   +-- ro state
              |       +-- ro ...
              |       .
              +-- rw ...
              |
              |
              +-- rw ...
              ...

rpcs:
   +-- x mpls-ldp-rpc
   +-- x . . . . .

notifications:
   +--- n mpls-ldp-notif
   +--- n ...
```

Figure 1

Before going into data model details, it is important to take note of the following points:

- This module aims to address only the core LDP/mLDP parameters as per RFC specification, as well as some widely used and deployed non-RFC features (such as label policies, session authentication etc). Any vendor specific feature should be defined in a vendor-specific augmentation of this model.

- Multi-topology LDP [RFC7307] and Multi-topology mLDP [I-D.iwijnand-mpls-mldp-multi-topology] are beyond the scope of this document.

- This module does not cover any applications running on top of LDP and mLDP, nor does it cover any OAM procedures for LDP and mLDP.

- This model is a VPN Forwarding and Routing (VRF)-centric model. It is important to note that [RFC4364] defines VRF tables and default forwarding tables as different, however from a yang modelling perspective this introduces unnecessary complications,

hence we are treating the default forwarding table as just another VRF.

- A "network-instance" as defined in [I-D.rtgyangdt-rtgwg-ni-model] refers to a VRF instance (both default and non-default) within the scope of this model.

- This model supports two address-families, namely "ipv4" and "ipv6".

- This model assumes platform-wide label space (i.e. label space Id of zero). However, when Upstream Label assignment [RFC6389] is in use, an upstream assigned label is looked up in a Context-Specific label space as defined in [RFC5331].

- The label and peer policies (including filters) are defined using a prefix-list. When used for a peer policy, the prefix refers to the LSR Id of the peer. The prefix-list is referenced from routing-policy model as defined in [I-D.ietf-rtgwg-policy-model].

- The use of grouping (templates) for bundling and grouping the configuration items is not employed in current revision, and is a subject for consideration in future.

- This model uses the terms LDP "neighbor"/"adjacency", "session", and "peer" with the following semantics:

  * Neighbor/Adjacency: An LDP enabled LSR that is discovered through LDP discovery mechanisms.

  * Session: An LDP neighbor with whom a TCP connection has been established.

  * Peer: An LDP session which has successfully progressed beyond its initialization phase and is either already exchanging the bindings or is ready to do so.

  It is to be noted that LDP Graceful Restart mechanisms defined in [RFC3478] allow keeping the exchanged bindings for some time after a session goes down with a peer. We call such a state -- i.e. keeping peer bindings without established or recovered peering -- a "stale" peer. When used in this document, the above terms will refer strictly to the semantics and definitions defined for them.

A graphical representation of LDP YANG data model is presented in Figure 3, Figure 5, Figure 11, and Figure 12. Whereas, the actual model definition in YANG is captured in Section 6.

While presenting the YANG tree view and actual .yang specification, this document assumes the reader is familiar with the concepts of YANG modeling, its presentation and its compilation.

3.2. Configuration

This specification defines the configuration parameters for base LDP as specified in [RFC5036] and LDP IPv6 [RFC7552]. Moreover, it incorporates provisions to enable LDP Capabilities [RFC5561], and defines some of the most significant and commonly used capabilities such as Typed Wildcard FEC [RFC5918], End-of-LIB [RFC5919], and LDP Upstream Label Assignment [RFC6389].

This specification supports VRF-centric configuration. For implementations that support protocol-centric configuration, with provision for inheritance and items that apply to all vrfs, we recommend an augmentation of this model such that any protocol-centric or all-vrf configuration is defined under their designated containers within the standard network-instance (please see Section 3.2.2)

This model augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg]. For LDP interfaces, this model refers the MPLS interface as defined under MPLS base specification [I-D.saad-mpls-base-yang]. Furthermore, as mentioned earlier, the configuration tree presents read-write intended configuration leave/items as well as read-only state of the applied configuration. The former is listed under "config" container and latter under "state" container.

Following is high-level configuration organization for LDP/mLDP:

```
       module: ietf-mpls-ldp
          +-- routing
           +-- control-plane-protocols
                 +-- mpls-ldp
                     +-- global
                     |    +-- ...
                     |    +-- ...
                     |    +-- address-family* [afi]
                     |        +-- . . .
                     |        +-- . . .
                     |    +-- discovery
                     |        +-- . . .
                     +-- peers
                          +-- ...
                          +-- ...
```

Figure 2

Given the configuration hierarchy, the model allows inheritance such that an item in a child tree is able to derive value from a similar or related item in one of the parent. For instance, hello holdtime can be configured per-VRF or per-VRF-interface, thus allowing inheritance as well flexibility to override with a different value at any child level.

Following is a simplified graphical representation of the data model for LDP configuration

```
   +--rw mpls-ldp!
      +--rw global
      |  +--rw config
      |  |  +--rw capability
      |  |  |  +--rw end-of-lib {capability-end-of-lib}?
      |  |  |  |  +--rw enable?   boolean
      |  |  |  +--rw typed-wildcard-fec {capability-typed-wildcard-fec}?
      |  |  |  |  +--rw enable?   boolean
      |  |  |  +--rw upstream-label-assignment {capability-upstream-label-assign
ment}?
      |  |  |     +--rw enable?   boolean
      |  |  +--rw graceful-restart
      |  |  |  +--rw enable?                boolean
      |  |  |  +--rw helper-enable?         boolean {graceful-restart-helper-mod
e}?
      |  |  |  +--rw reconnect-time?        uint16
      |  |  |  +--rw recovery-time?         uint16
      |  |  |  +--rw forwarding-holdtime?   uint16
      |  |  +--rw igp-synchronization-delay?   uint16
      |  |  +--rw lsr-id?                      yang:dotted-quad
```

```
      |  +--rw address-family* [afi]
      |  |  +--rw afi       ldp-address-family
      |  |  +--rw config
      |  |     +--rw enable?         boolean
      |  |     +--rw label-policy
      |  |     |  +--rw independent-mode
      |  |     |  |  +--rw assign {policy-label-assignment-config}?
      |  |     |  |  |  +--rw (prefix-option)?
      |  |     |  |  |     |  +--rw prefix-list?        prefix-list-ref
      |  |     |  |  |        +--rw host-routes-only?   boolean
      |  |     |  |  +--rw advertise
      |  |     |  |  |  +--rw explicit-null
      |  |     |  |  |  |  +--rw enable?        boolean
      |  |     |  |  |  |  +--rw prefix-list?   prefix-list-ref
      |  |     |  |  |  +--rw prefix-list?     prefix-list-ref
      |  |     |  |  +--rw accept
      |  |     |  |     +--rw prefix-list?   prefix-list-ref
      |  |     |  +--rw ordered-mode {policy-ordered-label-config}?
      |  |     |     +--rw egress-lsr
      |  |     |     |  +--rw prefix-list?   prefix-list-ref
      |  |     |     +--rw advertise
      |  |     |     |  +--rw prefix-list?   prefix-list-ref
      |  |     |     +--rw accept
      |  |     |        +--rw prefix-list?   prefix-list-ref
      |  |     +--rw ipv4
      |  |     |  +--rw transport-address?   inet:ipv4-address
      |  |     +--rw ipv6
      |  |        +--rw transport-address?   inet:ipv6-address
      |  +--rw discovery
      |  |  +--rw interfaces
      |  |  |  +--rw config
      |  |  |  |  +--rw hello-holdtime?   uint16
      |  |  |  |  +--rw hello-interval?   uint16
      |  |  |  +--rw interface* [interface]
      |  |  |     +--rw interface         mpls-interface-ref
      |  |  |     +--rw config
      |  |  |     |  +--rw hello-holdtime?              uint16
      |  |  |     |  +--rw hello-interval?              uint16
      |  |  |     |  +--rw igp-synchronization-delay?   uint16 {per-interface-ti
mer-config}?
      |  |  |     +--rw address-family* [afi]
      |  |  |        +--rw afi       ldp-address-family
      |  |  |        +--rw config
      |  |  |           +--rw enable?   boolean
      |  |  |           +--rw ipv4
      |  |  |           |  +--rw transport-address?   union
      |  |  |           +--rw ipv6
      |  |  |              +--rw transport-address?   union
      |  |  +--rw targeted
```

```
|  |     +--rw config
|  |     |  +--rw hello-holdtime?   uint16
|  |     |  +--rw hello-interval?   uint16
|  |     |  +--rw hello-accept {policy-extended-discovery-config}?
|  |     |     +--rw enable?          boolean
|  |     |     +--rw neighbor-list?   neighbor-list-ref
|  |     +--rw address-family* [afi]
|  |        +--rw afi      ldp-address-family
|  |        +--rw ipv4
|  |        |  +--rw target* [adjacent-address]
|  |        |     +--rw adjacent-address    inet:ipv4-address
|  |        |     +--rw config
|  |        |        +--rw enable?          boolean
|  |        |        +--rw local-address?   inet:ipv4-address
|  |        +--rw ipv6
|  |           +--rw target* [adjacent-address]
|  |              +--rw adjacent-address    inet:ipv6-address
|  |              +--rw config
|  |                 +--rw enable?          boolean
|  |                 +--rw local-address?   inet:ipv6-address
|  +--rw forwarding-nexthop {forwarding-nexthop-config}?
|  |  +--rw interfaces
|  |     +--rw interface* [interface]
|  |        +--rw interface         mpls-interface-ref
|  |        +--rw address-family* [afi]
|  |           +--rw afi       ldp-address-family
|  |           +--rw config
|  |              +--rw ldp-disable?    boolean
|  +--rw label-policy
|     +--rw independent-mode
|     |  +--rw assign {policy-label-assignment-config}?
|     |  |  +--rw (prefix-option)?
|     |  |        +--rw prefix-list?        prefix-list-ref
|     |  |        +--rw host-routes-only?   boolean
|     |  +--rw advertise
|     |  |  +--rw explicit-null
|     |  |  |  +--rw enable?        boolean
|     |  |  |  +--rw prefix-list?   prefix-list-ref
|     |  |  +--rw prefix-list?     prefix-list-ref
|     |  +--rw accept
|     |     +--rw prefix-list?   prefix-list-ref
|     +--rw ordered-mode {policy-ordered-label-config}?
|        +--rw egress-lsr
|        |  +--rw prefix-list?   prefix-list-ref
|        +--rw advertise
|        |  +--rw prefix-list?   prefix-list-ref
|        +--rw accept
|           +--rw prefix-list?   prefix-list-ref
```

```
      +--rw peers
         +--rw config
         |  +--rw session-authentication-md5-password?   string
         |  +--rw session-ka-holdtime?                   uint16
         |  +--rw session-ka-interval?                   uint16
         |  +--rw session-downstream-on-demand {session-downstream-on-demand-con
fig}?
         |     +--rw enable?      boolean
         |     +--rw peer-list?   peer-list-ref
         +--rw peer* [lsr-id]
            +--rw lsr-id    yang:dotted-quad
            +--rw config
               +--rw admin-down?                              boolean
               +--rw capability
               +--rw label-policy
               |  +--rw advertise
               |  |  +--rw prefix-list?   prefix-list-ref
               |  +--rw accept
               |     +--rw prefix-list?   prefix-list-ref
               +--rw session-authentication-md5-password?   string
               +--rw graceful-restart
               |  +--rw enable?           boolean
               |  +--rw reconnect-time?   uint16
               |  +--rw recovery-time?    uint16
               +--rw session-ka-holdtime?                   uint16
               +--rw session-ka-interval?                   uint16
               +--rw address-family
                  +--rw ipv4
                  |  +--rw label-policy
                  |     +--rw advertise
                  |     |  +--rw prefix-list?   prefix-list-ref
                  |     +--rw accept
                  |        +--rw prefix-list?   prefix-list-ref
                  +--rw ipv6
                     +--rw label-policy
                        +--rw advertise
                        |  +--rw prefix-list?   prefix-list-ref
                        +--rw accept
                           +--rw prefix-list?   prefix-list-ref
```

Figure 3

3.2.1. Configuration Hierarchy

The LDP configuration container is logically divided into following high-level config areas:

```
Per-VRF parameters
    o Global parameters
    o Per-address-family parameters
    o LDP Capabilities parameters
    o Hello Discovery parameters
        - interfaces
           - Per-interface:
              Global
              Per-address-family
        - targeted
           - Per-target
   o Peer parameters
        - Global
        - Per-peer
           Per-address-family
           Capabilities parameters
   o Forwarding parameters
```

Figure 4

Following subsections briefly explain these configuration areas.

3.2.1.1. Per-VRF parameters

LDP module resides under an network-instance and the scope of any LDP configuration defined under this tree is per network-instance (per-VRF). This configuration is further divided into sub categories as follows.

3.2.1.1.1. Per-VRF global parameters

There are configuration items that are available directly under a VRF instance and do not fall under any other sub tree. Example of such a parameter is LDP LSR id that is typically configured per VRF. To keep legacy LDP features and applications working in an LDP IPv4 networks with this model, this document recommends an operator to pick a routable IPv4 unicast address as an LSR Id.

3.2.1.1.2. Per-VRF Capabilities parameters

This container falls under global tree and holds the LDP capabilities that are to be enabled for certain features. By default, an LDP capability is disabled unless explicitly enabled. These capabilities are typically used to negotiate with LDP peer(s) the support/non-support related to a feature and its parameters. The scope of a capability enabled under this container applies to all LDP peers in the given VRF instance. There is also a peer level capability

container that is provided to override a capability that is enabled/ specified at VRF level.

3.2.1.1.3. Per-VRF Per-Address-Family parameters

Any LDP configuration parameter related to IP address family (AF) whose scope is VRF wide is configured under this tree. The examples of per-AF parameters include enabling LDP for an address family, prefix-list based label policies, and LDP transport address.

3.2.1.1.4. Per-VRF Hello Discovery parameters

This container is used to hold LDP configuration related to Hello and discovery process for both basic (link) and extended (targeted) discovery.

The "interfaces" is a container to configure parameters related to VRF interfaces. There are parameters that apply to all interfaces (such as hello timers), as well as parameters that can be configured per-interface. Hence, an interface list is defined under "interfaces" container. The model defines parameters to configure per-interface non AF related items, as well as per-interface per-AF items. The example of former is interface hello timers, and example of latter is enabling hellos for a given AF under an interface.

The "targeted" container under a VRF instance allows to configure LDP targeted discovery related parameters. Within this container, the "target" list provides a mean to configure multiple target addresses to perform extended discovery to a specific destination target, as well as to fine-tune the per-target parameters.

3.2.1.1.5. Per-VRF Peer parameters

This container is used to hold LDP configuration related to LDP sessions and peers under a VRF instance. This container allows to configure parameters that either apply on VRF’s all peers or a subset (peer-list) of VRF peers. The example of such parameters include authentication password, session KA timers etc. Moreover, the model also allows per-peer parameter tuning by specifying a "peer" list under the "peers" container. A peer is uniquely identified using its LSR Id and hence LSR Id is the key for peer list

Like per-interface parameters, some per-peer parameters are AF-agnostic (i.e. either non AF related or apply to both IP address families), and some that belong to an AF. The example of former is per-peer session password configuration, whereas the example of latter is prefix-list based label policies (inbound and outbound) that apply to a given peer.

3.2.1.1.6. Per-VRF Forwarding parameters

This container is used to hold configuration used to control LDP forwarding behavior under a VRF instance. One example of a configuration under this container is when a user wishes to enable neighbor discovery on an interface but wishes to disable use of the same interface as forwarding nexthop. This example configuration makes sense only when there are more than one LDP enabled interfaces towards the neighbor.

3.2.2. All-VRFs Configuration

[Ed note: TODO]

3.3. Operational State

Operational state of LDP can be queried and obtained from read-only state containers that fall under the same tree (/rt:routing/ rt:control-plane-protocols/) as the configuration.

Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the protocol. [Ed note: This is where this model differs presently from [I-D.openconfig-netmod-opstate] and subject to alignment in later revisions]

Following is a simplified graphical representation of the data model for LDP operational state.

```
module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
   +--rw mpls-ldp!
      +--rw global
      |  +--ro state
      |  |  +--ro capability
      |  |  |  +--ro end-of-lib {capability-end-of-lib}?
      |  |  |  |  +--ro enable?   boolean
      |  |  |  +--ro typed-wildcard-fec {capability-typed-wildcard-fec}?
      |  |  |  |  +--ro enable?   boolean
      |  |  |  +--ro upstream-label-assignment {capability-upstream-label-assign
ment}?
      |  |  |  |  +--ro enable?   boolean
      |  |  +--ro graceful-restart
      |  |  |  +--ro enable?                boolean
      |  |  |  +--ro helper-enable?         boolean {graceful-restart-helper-mod
e}?
      |  |  |  +--ro reconnect-time?        uint16
      |  |  |  +--ro recovery-time?         uint16
      |  |  |  +--ro forwarding-holdtime?   uint16
```

```
|  |  +--ro igp-synchronization-delay?   uint16
|  |  +--ro lsr-id?                      yang:dotted-quad
|  +--rw address-family* [afi]
|  |  +--rw afi       ldp-address-family
|  |  +--ro state
|  |     +--ro enable?         boolean
|  |     +--ro label-policy
|  |     |  +--ro independent-mode
|  |     |  |  +--ro assign {policy-label-assignment-config}?
|  |     |  |  |  +--ro (prefix-option)?
|  |     |  |  |     +--:(prefix-list)
|  |     |  |  |     |  +--ro prefix-list?        prefix-list-ref
|  |     |  |  |     +--:(host-routes-only)
|  |     |  |  |        +--ro host-routes-only?   boolean
|  |     |  |  +--ro advertise
|  |     |  |  |  +--ro explicit-null
|  |     |  |  |  |  +--ro enable?        boolean
|  |     |  |  |  |  +--ro prefix-list?   prefix-list-ref
|  |     |  |  |  +--ro prefix-list?     prefix-list-ref
|  |     |  |  +--ro accept
|  |     |  |     +--ro prefix-list?   prefix-list-ref
|  |     |  +--ro ordered-mode {policy-ordered-label-config}?
|  |     |     +--ro egress-lsr
|  |     |     |  +--ro prefix-list?   prefix-list-ref
|  |     |     +--ro advertise
|  |     |     |  +--ro prefix-list?   prefix-list-ref
|  |     |     +--ro accept
|  |     |        +--ro prefix-list?   prefix-list-ref
|  |     +--ro ipv4
|  |     |  +--ro transport-address?   inet:ipv4-address
|  |     |  +--ro bindings
|  |     |     +--ro address* [address]
|  |     |     |  +--ro address      inet:ipv4-address
|  |     |     |  +--ro advertisement-type?   advertised-received
|  |     |     |  +--ro peer?        leafref
|  |     |     +--ro fec-label* [fec]
|  |     |        +--ro fec     inet:ipv4-prefix
|  |     |        +--ro peer* [peer advertisement-type]
|  |     |           +--ro peer                  leafref
|  |     |           +--ro advertisement-type    advertised-received
|  |     |           +--ro label?                mpls:mpls-label
|  |     |           +--ro used-in-forwarding?   boolean
|  |     +--ro ipv6
|  |        +--ro transport-address?   inet:ipv6-address
|  |        +--ro binding
|  |           +--ro address* [address]
|  |           |  +--ro address      inet:ipv6-address
|  |           |  +--ro advertisement-type?   advertised-received
```

```
      |  |           |  +--ro peer?        leafref
      |  |           +--ro fec-label* [fec]
      |  |              +--ro fec     inet:ipv6-prefix
      |  |              +--ro peer* [peer advertisement-type]
      |  |                 +--ro peer                  leafref
      |  |                 +--ro advertisement-type    advertised-received
      |  |                 +--ro label?                mpls:mpls-label
      |  |                 +--ro used-in-forwarding?   boolean
      |  +--rw discovery
      |  |  +--rw interfaces
      |  |  |  +--ro state
      |  |  |  |  +--ro hello-holdtime?   uint16
      |  |  |  |  +--ro hello-interval?   uint16
      |  |  |  +--rw interface* [interface]
      |  |  |     +--ro state
      |  |  |     |  +--ro hello-holdtime?              uint16
      |  |  |     |  +--ro hello-interval?              uint16
      |  |  |     |  +--ro igp-synchronization-delay?   uint16 {per-interface-ti
mer-config}?
      |  |  |     |  +--ro next-hello?                  uint16
      |  |  |     +--rw address-family* [afi]
      |  |  |        +--rw afi       ldp-address-family
      |  |  |        +--ro state
      |  |  |           +--ro enable?   boolean
      |  |  |           +--ro ipv4
      |  |  |           |  +--ro transport-address?   union
      |  |  |           |  +--ro hello-adjacencies* [adjacent-address]
      |  |  |           |     +--ro adjacent-address    inet:ipv4-address
      |  |  |           |     +--ro flag*               identityref
      |  |  |           |     +--ro hello-holdtime
      |  |  |           |     |  +--ro adjacent?     uint16
      |  |  |           |     |  +--ro negotiated?   uint16
      |  |  |           |     |  +--ro remaining?    uint16
      |  |  |           |     +--ro next-hello?         uint16
      |  |  |           |     +--ro statistics
      |  |  |           |     |  +--ro discontinuity-time    yang:date-and-time
      |  |  |           |     |  +--ro hello-received?       yang:counter64
      |  |  |           |     |  +--ro hello-dropped?        yang:counter64
      |  |  |           |     +--ro peer?               leafref
      |  |  |           +--ro ipv6
      |  |  |              +--ro transport-address?   union
      |  |  |              +--ro hello-adjacencies* [adjacent-address]
      |  |  |                 +--ro adjacent-address    inet:ipv6-address
      |  |  |                 +--ro flag*               identityref
      |  |  |                 +--ro hello-holdtime
      |  |  |                 |  +--ro adjacent?     uint16
      |  |  |                 |  +--ro negotiated?   uint16
      |  |  |                 |  +--ro remaining?    uint16
      |  |  |                 +--ro next-hello?         uint16
```

```
      |  |  |                +--ro statistics
      |  |  |                |  +--ro discontinuity-time    yang:date-and-time
      |  |  |                |  +--ro hello-received?       yang:counter64
      |  |  |                |  +--ro hello-dropped?        yang:counter64
      |  |  |                +--ro peer?               leafref
      |  |  +--rw targeted
      |  |     +--ro state
      |  |     |  +--ro hello-holdtime?   uint16
      |  |     |  +--ro hello-interval?   uint16
      |  |     |  +--ro hello-accept {policy-extended-discovery-config}?
      |  |     |     +--ro enable?          boolean
      |  |     |     +--ro neighbor-list?   neighbor-list-ref
      |  |     +--rw address-family* [afi]
      |  |        +--rw afi      ldp-address-family
      |  |        +--ro state
      |  |        |  +--ro ipv4
      |  |        |  |  +--ro hello-adjacencies* [local-address adjacent-address
]
      |  |        |  |     +--ro local-address       inet:ipv4-address
      |  |        |  |     +--ro adjacent-address    inet:ipv4-address
      |  |        |  |     +--ro flag*               identityref
      |  |        |  |     +--ro hello-holdtime
      |  |        |  |     |  +--ro adjacent?     uint16
      |  |        |  |     |  +--ro negotiated?   uint16
      |  |        |  |     |  +--ro remaining?    uint16
      |  |        |  |     +--ro next-hello?         uint16
      |  |        |  |     +--ro statistics
      |  |        |  |     |  +--ro discontinuity-time    yang:date-and-time
      |  |        |  |     |  +--ro hello-received?       yang:counter64
      |  |        |  |     |  +--ro hello-dropped?        yang:counter64
      |  |        |  |     +--ro peer?               leafref
      |  |        |  +--ro ipv6
      |  |        |     +--ro hello-adjacencies* [local-address adjacent-address
]
      |  |        |        +--ro local-address       inet:ipv6-address
      |  |        |        +--ro adjacent-address    inet:ipv6-address
      |  |        |        +--ro flag*               identityref
      |  |        |        +--ro hello-holdtime
      |  |        |        |  +--ro adjacent?     uint16
      |  |        |        |  +--ro negotiated?   uint16
      |  |        |        |  +--ro remaining?    uint16
      |  |        |        +--ro next-hello?         uint16
      |  |        |        +--ro statistics
      |  |        |        |  +--ro discontinuity-time    yang:date-and-time
      |  |        |        |  +--ro hello-received?       yang:counter64
      |  |        |        |  +--ro hello-dropped?        yang:counter64
      |  |        |        +--ro peer?               leafref
      |  |        +--rw ipv4
      |  |        |  +--rw target* [adjacent-address]
      |  |        |     +--rw adjacent-address    inet:ipv4-address
```

```
      |  |        |     +--ro state
      |  |        |        +--ro enable?          boolean
      |  |        |        +--ro local-address?   inet:ipv4-address
      |  |        +--rw ipv6
      |  |           +--rw target* [adjacent-address]
      |  |              +--rw adjacent-address    inet:ipv6-address
      |  |              +--ro state
      |  |                 +--ro enable?          boolean
      |  |                 +--ro local-address?   inet:ipv6-address
      |  +--rw forwarding-nexthop {forwarding-nexthop-config}?
      |  |  +--rw interfaces
      |  |     +--rw interface* [interface]
      |  |        +--rw interface         mpls-interface-ref
      |  |        +--rw address-family* [afi]
      |  |           +--rw afi       ldp-address-family
      |  |           +--ro state
      |  |              +--ro ldp-disable?    boolean
      +--rw peers
         +--ro state
         |  +--ro session-authentication-md5-password?   string
         |  +--ro session-ka-holdtime?                   uint16
         |  +--ro session-ka-interval?                   uint16
         |  +--ro session-downstream-on-demand {session-downstream-on-demand-con
fig}?
         |     +--ro enable?      boolean
         |     +--ro peer-list?   peer-list-ref
         +--rw peer* [lsr-id]
            +--rw lsr-id    yang:dotted-quad
            +--ro state
               +--ro admin-down?                              boolean
               +--ro capability
               +--ro label-policy
               |  +--ro advertise
               |  |  +--ro prefix-list?   prefix-list-ref
               |  +--ro accept
               |     +--ro prefix-list?   prefix-list-ref
               +--ro session-authentication-md5-password?   string
               +--ro graceful-restart
               |  +--ro enable?           boolean
               |  +--ro reconnect-time?   uint16
               |  +--ro recovery-time?    uint16
               +--ro session-ka-holdtime?                   uint16
               +--ro session-ka-interval?                   uint16
               +--ro address-family
               |  +--ro ipv4
               |  |  +--ro label-policy
               |  |  |  +--ro advertise
               |  |  |  |  +--ro prefix-list?   prefix-list-ref
               |  |  |  +--ro accept
```

```
|  |  |     +--ro prefix-list?   prefix-list-ref
|  |  +--ro hello-adjacencies* [local-address adjacent-address]
|  |     +--ro local-address       inet:ipv4-address
|  |     +--ro adjacent-address    inet:ipv4-address
|  |     +--ro flag*               identityref
|  |     +--ro hello-holdtime
|  |     |  +--ro adjacent?     uint16
|  |     |  +--ro negotiated?   uint16
|  |     |  +--ro remaining?    uint16
|  |     +--ro next-hello?         uint16
|  |     +--ro statistics
|  |     |  +--ro discontinuity-time    yang:date-and-time
|  |     |  +--ro hello-received?       yang:counter64
|  |     |  +--ro hello-dropped?        yang:counter64
|  |     +--ro interface?          mpls-interface-ref
|  +--ro ipv6
|     +--ro label-policy
|     |  +--ro advertise
|     |  |  +--ro prefix-list?   prefix-list-ref
|     |  +--ro accept
|     |     +--ro prefix-list?   prefix-list-ref
|     +--ro hello-adjacencies* [local-address adjacent-address]
|        +--ro local-address       inet:ipv6-address
|        +--ro adjacent-address    inet:ipv6-address
|        +--ro flag*               identityref
|        +--ro hello-holdtime
|        |  +--ro adjacent?     uint16
|        |  +--ro negotiated?   uint16
|        |  +--ro remaining?    uint16
|        +--ro next-hello?         uint16
|        +--ro statistics
|        |  +--ro discontinuity-time    yang:date-and-time
|        |  +--ro hello-received?       yang:counter64
|        |  +--ro hello-dropped?        yang:counter64
|        +--ro interface?          mpls-interface-ref
+--ro label-advertisement-mode
|  +--ro local?        label-adv-mode
|  +--ro peer?         label-adv-mode
|  +--ro negotiated?   label-adv-mode
+--ro next-keep-alive?                       uint16
+--ro peer-ldp-id?                           yang:dotted-quad
+--ro received-peer-state
|  +--ro graceful-restart
|  |  +--ro enable?           boolean
|  |  +--ro reconnect-time?   uint16
|  |  +--ro recovery-time?    uint16
|  +--ro capability
|     +--ro end-of-lib
```

```
|     |  +--ro enable?   boolean
|     +--ro typed-wildcard-fec
|     |  +--ro enable?   boolean
|     +--ro upstream-label-assignment
|        +--ro enable?   boolean
+--ro session-holdtime
|  +--ro peer?         uint16
|  +--ro negotiated?   uint16
|  +--ro remaining?    uint16
+--ro session-state?                          enumeration
+--ro tcp-connection
|  +--ro local-address?    inet:ip-address
|  +--ro local-port?       inet:port-number
|  +--ro remote-address?   inet:ip-address
|  +--ro remote-port?      inet:port-number
+--ro up-time?                                string
+--ro statistics
   +--ro discontinuity-time          yang:date-and-time
   +--ro received
   |  +--ro total-octets?          yang:counter64
   |  +--ro total-messages?        yang:counter64
   |  +--ro address?               yang:counter64
   |  +--ro address-withdraw?      yang:counter64
   |  +--ro initialization?        yang:counter64
   |  +--ro keepalive?             yang:counter64
   |  +--ro label-abort-request?   yang:counter64
   |  +--ro label-mapping?         yang:counter64
   |  +--ro label-release?         yang:counter64
   |  +--ro label-request?         yang:counter64
   |  +--ro label-withdraw?        yang:counter64
   |  +--ro notification?          yang:counter64
   +--ro sent
   |  +--ro total-octets?          yang:counter64
   |  +--ro total-messages?        yang:counter64
   |  +--ro address?               yang:counter64
   |  +--ro address-withdraw?      yang:counter64
   |  +--ro initialization?        yang:counter64
   |  +--ro keepalive?             yang:counter64
   |  +--ro label-abort-request?   yang:counter64
   |  +--ro label-mapping?         yang:counter64
   |  +--ro label-release?         yang:counter64
   |  +--ro label-request?         yang:counter64
   |  +--ro label-withdraw?        yang:counter64
   |  +--ro notification?          yang:counter64
   +--ro total-addresses?            uint32
   +--ro total-labels?               uint32
   +--ro total-fec-label-bindings?   uint32
```

Figure 5

3.3.1. Derived States

Following are main areas for which LDP operational "derived" state is defined:

Neighbor Adjacencies

Peer

Bindings (FEC-label and address)

Capabilities

3.3.1.1. Adjacency state

Neighbor adjacencies are per address-family hello adjacencies that are formed with neighbors as result of LDP basic or extended discovery. In terms of organization, there is a source of discovery (e.g. interface or target address) along with its associated parameters and one or more discovered neighbors along with neighbor discovery related parameters. For the basic discovery, there could be more than one discovered neighbor for a given source (interface), whereas there is at most one discovered neighbor for an extended discovery source (local-address and target-address). This is also to be noted that the reason for a targeted neighbor adjacency could be either an active source (locally configured targeted) or passive source (to allow any incoming extended/targeted hellos). A neighbor/adjacency record also contains session-state that helps highlight whether a given adjacency has progressed to subsequent session level or to eventual peer level.

Following captures high level tree hierarchy for neighbor adjacency state.

```
   +--rw mpls-ldp!
      +--rw discovery
         +--rw interfaces
         |  +--rw interface* [interface]
         |     +--rw address-family* [af]
         |        +--ro state
         |           +--ro ipv4 (or ipv6)
         |              +--ro hello-adjacencies* [adjacent-address]
         |                 +--ro adjacent-address
         |                    . . . .
         |                    . . . .
         +--rw targeted
            +--rw address-family* [afi]
                +--rw afi      address-family
                   +--ro state
                      +--ro ipv4 (or ipv6)
                         +--ro hello-adjacencies* [local-address adjacent-addres
s]
                            +--ro local-address
                            +--ro adjacent-address
                               . . . .
                               . . . .
```

Figure 6

3.3.1.2. Peer state

Peer related derived state is presented under peers tree. This is one of the core state that provides info on the session related parameters (mode, authentication, KA timeout etc.), TCP connection info, hello adjacencies for the peer, statistics related to messages and bindings, and capabilities exchange info.

Following captures high level tree hierarchy for peer state.

```
  +--rw mpls-ldp!
      +--rw peers
         +--rw peer* [lsr-id]
            +--rw lsr-id
            +--ro state
               +--ro session-ka-holdtime?
               +-- . . . .
               +-- . . . .
               +--ro capability
               +  +ro -- . . .
               +--ro address-family
               |  +--ro ipv4 (or ipv6)
               |     +--ro hello-adjacencies* [local-address adjacent-address]
               |        . . . .
               |        . . . .
               +--ro received-peer-state
               |  +--ro . . . .
               |  +--ro capability
               |     +--ro . . . .
               +--ro statistics
                  +-- . . . .
                  +-- . . . .
```

Figure 7

3.3.1.3. Bindings state

Binding state provides information on LDP FEC-label bindings as well as address binding for both inbound (received) as well as outbound (advertised) direction. FEC-label bindings are presented as a FEC-centric view, and address bindings are presented as an address-centric view:

```
        FEC-Label bindings:
            FEC 200.1.1.1/32:
              advertised: local-label 16000
                peer 192.168.0.2:0
                peer 192.168.0.3:0
                peer 192.168.0.4:0
              received:
                peer 192.168.0.2:0, label 16002, used-in-forwarding=Yes
                peer 192.168.0.3:0, label 17002, used-in-forwarding=No
            FEC 200.1.1.2/32:
               . . . .
            FEC 201.1.0.0/16:
               . . . .


        Address bindings:
            Addr 1.1.1.1:
              advertised
            Addr 1.1.1.2:
              advertised
            Addr 2.2.2.2:
              received, peer 192.168.0.2
            Addr 2.2.2.22:
              received, peer 192.168.0.2
            Addr 3.3.3.3:
              received, peer 192.168.0.3
            Addr 3.3.3.33:
              received, peer 192.168.0.3
```

Figure 8

Note that all local addresses are advertised to all peers and hence no need to provide per-peer information for local address advertisement. Furthermore, note that it is easy to derive a peer-centric view for the bindings from the information already provided in this model.

Following captures high level tree hierarchy for bindings state.

```
   +--rw mpls-ldp!
      +--rw global
         +--rw address-family* [afi]
            +--rw afi       address-family
            +--ro state
               +--ro ipv4 (or ipv6)
                  +--ro bindings
                     +--ro address* [address]
                     |  +--ro address
                     |  +--ro direction?   advertised-received
                     |  +--ro peer?        leafref
                     +--ro fec-label* [fec]
                        +--ro fec     inet:ipv4-prefix
                        +--ro peer* [peer advertisement-type]
                           +--ro peer                  leafref
                           +--ro advertisement-type    advertised-received
                           +--ro label?                mpls:mpls-label
                           +--ro used-in-forwarding?   boolean
```

Figure 9

3.3.1.4. Capabilities state

LDP capabilities state comprise two types of information - global information (such as timer etc.), and per-peer information.

Following captures high level tree hierarchy for LDP capabilities state.

```
   +--rw mpls-ldp!
      +--rw global
      |  +--ro state
      |     +--ro capability
      |        +--ro . . . .
      |        +--ro . . . .
      +--rw peers
         +--rw peer* [lsr-id]
            +--rw lsr-id    yang:dotted-quad
            +--ro state
               +--ro received-peer-state
                  +--ro capability
                     +--ro . . . .
                     +--ro . . . .
```

Figure 10

3.4. Notifications

This model defines a list of notifications to inform client of important events detected during the protocol operation. These events include events related to changes in the operational state of an LDP peer, hello adjacency, and FEC etc. It is to be noted that an LDP FEC is treated as operational (up) as long as it has at least 1 NHLFE with outgoing label.

Following is a simplified graphical representation of the data model for LDP notifications.

```
module: ietf-mpls-ldp
notifications:
   +---n mpls-ldp-peer-event
   |  +--ro event-type?   oper-status-event-type
   |  +--ro peer-ref?     leafref
   +---n mpls-ldp-hello-adjacency-event
   |  +--ro event-type?   oper-status-event-type
   |  +--ro (hello-adjacency-type)?
   |     +--:(targeted)
   |     |  +--ro targeted
   |     |     +--ro target-address?   inet:ip-address
   |     +--:(link)
   |        +--ro link
   |           +--ro next-hop-interface?   mpls-interface-ref
   |           +--ro next-hop-address?     inet:ip-address
   +---n mpls-ldp-fec-event
      +--ro event-type?   oper-status-event-type
      +--ro prefix?       inet:ip-prefix
```

Figure 11

3.5. Actions

This model defines a list of rpcs that allow performing an action or executing a command on the protocol. For example, it allows to clear (reset) LDP peers, hello-adjacencies, and statistics. The model makes an effort to provide different level of control so that a user is able to either clear all, or clear all for a given type, or clear a specific entity.

Following is a simplified graphical representation of the data model for LDP actions.

```
module: ietf-mpls-ldp
rpcs:
   +---x mpls-ldp-clear-peer
   |  +---w input
   |     +---w lsr-id?   union
   +---x mpls-ldp-clear-hello-adjacency
   |  +---w input
   |     +---w hello-adjacency
   |        +---w (hello-adjacency-type)?
   |           +--:(targeted)
   |           |  +---w targeted!
   |           |     +---w target-address?   inet:ip-address
   |           +--:(link)
   |              +---w link!
   |                 +---w next-hop-interface?   mpls-interface-ref
   |                 +---w next-hop-address?     inet:ip-address
   +---x mpls-ldp-clear-peer-statistics
      +---w input
         +---w lsr-id?   union
```

Figure 12

4. mLDP YANG Model

4.1. Overview

Due to tight dependency of mLDP on LDP, mLDP model builds on top of LDP model defined earlier in the document. Following are the main mLDP areas and documents that are within the scope of this model:

- mLDP Base Specification [RFC6388]
- mLDP Recursive FEC [RFC6512]
- Targeted mLDP [RFC7060]
- mLDP Fast-Reroute (FRR)
  - Node Protection [RFC7715]
  - Multicast-only
- Hub-and-Spoke Multipoint LSPs [RFC7140]
- mLDP In-band Signaling [RFC6826] (future revision)
- mLDP In-band signaling in a VRF [RFC7246]

- o mLDP In-band Signaling with Wildcards [RFC7438] (future revision)
- o Configured Leaf LSPs (manually provisioned)

[Ed Note: Some of the topics in the above list are to be addressed/ added in later revision of this document].

4.2. Configuration

4.2.1. Configuration Hierarchy

In terms of overall configuration layout, following figure highlights extensions to LDP configuration model to incorporate mLDP:

```
      +-- mpls-ldp
         +-- ...
         +-- ...
         +-- mldp
         |  +-- ...
         |  +-- ...
         |  +-- address-family* [af]
         |     +-- af
         |        +-- ...
         |        +-- ...
         +-- global
         |  +-- ...
         |  +-- capability
         |     +-- ...
         |     +-- ...
         |     +-- mldp
         |        +-- ...
         |        +-- ...
         +-- discovery
         |  +-- ...
         |  +-- ...
         +-- forwarding-nexthop
         |  +-- interfaces
         |     +-- interface* [interface]
         |        +-- interface
         |        +-- address-family* [af]
         |           +-- af
         |           +-- ...
         |           +-- mldp-disable
         +-- peers
            +-- ...
            +-- ...
            +-- peer* [lsr-id]
                +-- ...
                +-- ...
                +-- capability
                    +-- ...
                    +-- ...
                    +-- mldp
                        +-- ...
                        +-- ...
```

Figure 13

From above hierarchy, we can categorize mLDP configuration parameters into two types:

- Parameters that leverage/extend LDP containers and parameters
- Parameters that are mLDP specific

Following subsections first describe mLDP specific configuration parameters, followed by those leveraging LDP.

4.2.2. mldp container

mldp container resides directly under "mpls-ldp" and holds the configuration related to items that are mLDP specific. The main items under this container are:

- mLDP enabling: To enable mLDP under a (VRF) routing instance, mldp container is enabled under LDP. Given that mLDP requires LDP signalling, it is not sensible to allow disabling LDP control plane under a (VRF) network-instance while requiring mLDP to be enabled for the same. However, if a user wishes only to allow signalling for multipoint FECs on an LDP/mLDP enabled VRF instance, he/she can use LDP label-policies to disable unicast FECs under the VRF.
- mLDP per-AF features: mLDP manages its own list of IP address-families and the features enabled underneath. The per-AF mLDP configuration items include:
  - Multicast-only FRR: This enables Multicast-only FRR functionality for a given AF under mLDP. The feature allows route-policy to be configured for finer control/applicability of the feature.
  - Recursive FEC: The recursive-fec feature [RFC6512] can be enabled per AF with a route-policy.
  - Configured Leaf LSPs: To provision multipoint leaf LSP manually, a container is provided per-AF under LDP. The configuration is flexible and allows a user to specify MP LSPs of type p2mp or mp2mp with IPv4 or IPv6 root address(es) by using either LSP-Id or (S,G).

Targeted mLDP feature specification [RFC7060] do not require any mLDP specific configuration. It, however, requires LDP upstream-label-assignment capability [RFC6389] to be enabled.

4.2.3. Leveraging LDP containers

mLDP configuration model leverages following configuration areas and containers that are already defined for LDP:

- Capabilities: A new container "mldp" is defined under Capabilities container. This new container specifies any mLDP specific capabilities and their parameters. Moreover, a new "mldp" container is also added under per-peer capability container to override/control mLDP specific capabilities on a peer level. In the scope of this document, the most important capabilities related to mLDP are p2mp, mp2mp, make-before-break, hub-and-spoke, and node-protection.

- Discovery and Peer: mLDP requires LDP discovery and peer procedures to form mLDP peering. A peer is treated as mLDP peer only when either P2MP or MP2MP capabilities have been successfully exchanged with the peer. If a user wish to selectively enable or disable mLDP with a LDP-enabled peer, he/she may use per-peer mLDP capabilities configuration. [Ed Note: The option to control mLDP enabling/disabling on a peer-list is being explored for future]. In most common deployments, it is desirable to disable mLDP (capabilities announcements) on a targeted-only LDP peering, where targeted-only peer is the one whose discovery sources are targeted only. In future revision, a configuration option for this support will also be provided.

- Forwarding: By default, mLDP is allowed to select any of the LDP enabled interface as a downstream interface towards a nexthop (LDP/mLDP peer) for MP LSP programming. However, a configuration option is provided to allow mLDP to exclude a given interface from such a selection. Note that such a configuration option will be useful only when there are more than one interfaces available for the downstream selection.

This goes without saying that mLDP configuration tree follows the same approach as LDP, where the tree comprise leafs for intended configuration.

4.2.4. YANG tree

The following figure captures the YANG tree for mLDP configuration. To keep the focus, the figure has been simplified to display only mLDP items without any LDP items.

```
module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
   +--rw mpls-ldp!
```

```
      +--rw global
      |  +--rw config
      |  |  +--rw capability
      |  |     +--rw mldp {mldp}?
      |  |        +--rw p2mp
      |  |        |  +--rw enable?   boolean
      |  |        +--rw mp2mp
      |  |        |  +--rw enable?   boolean
      |  |        +--rw make-before-break
      |  |        |  +--rw enable?             boolean
      |  |        |  +--rw switchover-delay?   uint16
      |  |        |  +--rw timeout?            uint16
      |  |        +--rw hub-and-spoke {capability-mldp-hsmp}?
      |  |        |  +--rw enable?   boolean
      |  |        +--rw node-protection {capability-mldp-node-protection}?
      |  |           +--rw plr?           boolean
      |  |           +--rw merge-point
      |  |              +--rw enable?                                boolean
      |  |              +--rw targeted-session-teardown-delay?   uint16
      |  +--rw mldp {mldp}?
      |  |  +--rw config
      |  |  |  +--rw enable?   boolean
      |  |  +--rw address-family* [afi]
      |  |     +--rw afi                      ldp-address-family
      |  |     +--rw config
      |  |     |  +--rw multicast-only-frr {mldp-mofrr}?
      |  |     |  |  +--rw prefix-list?   prefix-list-ref
      |  |     |  +--rw recursive-fec
      |  |     |     +--rw prefix-list?   prefix-list-ref
      |  |     +--rw configured-leaf-lsps
      |  |        +--rw p2mp
      |  |        |  +--rw roots-ipv4
      |  |        |  |  +--rw root* [root-address]
      |  |        |  |     +--rw root-address    inet:ipv4-address
      |  |        |  |     +--rw lsp* [lsp-id source-address group-address]
      |  |        |  |        +--rw lsp-id            uint16
      |  |        |  |        +--rw source-address    inet:ipv4-address
      |  |        |  |        +--rw group-address     inet:ipv4-address-no-zone
      |  |        |  +--rw roots-ipv6
      |  |        |     +--rw root* [root-address]
      |  |        |        +--rw root-address    inet:ipv6-address
      |  |        |        +--rw lsp* [lsp-id source-address group-address]
      |  |        |           +--rw lsp-id            uint16
      |  |        |           +--rw source-address    inet:ipv6-address
      |  |        |           +--rw group-address     inet:ipv6-address-no-zone
      |  |        +--rw mp2mp
      |  |           +--rw roots-ipv4
      |  |           |  +--rw root* [root-address]
```

```
|  |           |     +--rw root-address    inet:ipv4-address
|  |           |     +--rw lsp* [lsp-id source-address group-address]
|  |           |        +--rw lsp-id            uint16
|  |           |        +--rw source-address    inet:ipv4-address
|  |           |        +--rw group-address     inet:ipv4-address-no-zone
|  |           +--rw roots-ipv6
|  |              +--rw root* [root-address]
|  |                 +--rw root-address    inet:ipv6-address
|  |                 +--rw lsp* [lsp-id source-address group-address]
|  |                    +--rw lsp-id            uint16
|  |                    +--rw source-address    inet:ipv6-address
|  |                    +--rw group-address     inet:ipv6-address-no-zone
|  +--rw forwarding-nexthop {forwarding-nexthop-config}?
|     +--rw interfaces
|        +--rw interface* [interface]
|           +--rw interface         if:interface-ref
|           +--rw address-family* [afi]
|              +--rw afi       address-family
|              +--rw config
|                 +--rw mldp-disable?   boolean {mldp}?
+--rw peers
   +--rw peer* [lsr-id]
         +--rw capability
            +--rw mldp {mldp}?
               +--rw p2mp
               |  +--rw enable?   boolean
               +--rw mp2mp
               |  +--rw enable?   boolean
               +--rw make-before-break
               |  +--rw enable?             boolean
               |  +--rw switchover-delay?   uint16
               |  +--rw timeout?            uint16
               +--rw hub-and-spoke {capability-mldp-hsmp}?
               |  +--rw enable?   boolean
                +--rw node-protection {capability-mldp-node-protection}?
                  +--rw plr?           boolean
                  +--rw merge-point
                     +--rw enable?                              boolean
                     +--rw targeted-session-teardown-delay?   uint16
```

Figure 14

4.3. Operational State

Operational state of mLDP can be queried and obtained from this read-only container "mldp" which resides under mpls-ldp container.

Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the mLDP protocol.

Following is a simplified graphical representation of the data model for mLDP operational state:

```
module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
   +--rw mpls-ldp!
      +--rw global
      |  +--ro state
      |  |  +--ro capability
      |  |     +--ro mldp {mldp}?
      |  |        +--ro p2mp
      |  |        |  +--ro enable?   boolean
      |  |        +--ro mp2mp
      |  |        |  +--ro enable?   boolean
      |  |        +--ro make-before-break
      |  |        |  +--ro enable?             boolean
      |  |        |  +--ro switchover-delay?   uint16
      |  |        |  +--ro timeout?            uint16
      |  |        +--ro hub-and-spoke {capability-mldp-hsmp}?
      |  |        |  +--ro enable?   boolean
      |  |        +--ro node-protection {capability-mldp-node-protection}?
      |  |           +--ro plr?           boolean
      |  |           +--ro merge-point
      |  |              +--ro enable?                              boolean
      |  |              +--ro targeted-session-teardown-delay?   uint16
      |  |
      |  +--rw mldp {mldp}?
      |  |  +--ro state
      |  |     +--ro enable?   boolean
      |  |  +--rw address-family* [afi]
      |  |     +--rw afi                     ldp-address-family
      |  |     +--ro state
      |  |        +--ro multicast-only-frr {mldp-mofrr}?
      |  |        |  +--ro prefix-list?   prefix-list-ref
      |  |        +--ro recursive-fec
      |  |        |  +--ro prefix-list?   prefix-list-ref
      |  |        +--ro ipv4
      |  |        |  +--ro roots
      |  |        |  |  +--ro root* [root-address]
      |  |        |  |     +--ro root-address    inet:ipv4-address
      |  |        |  |     +--ro is-self?        boolean
      |  |        |  |     +--ro reachability* [address interface]
      |  |        |  |        +--ro address      inet:ipv4-address
      |  |        |  |        +--ro interface    mpls-interface-ref
```

```
      |  |        |  |        +--ro peer?        leafref
      |  |        |  +--ro bindings
      |  |        |     +--ro opaque-type-lspid
      |  |        |     |  +--ro fec-label* [root-address lsp-id recur-root-addr
ess recur-rd]
      |  |        |     |     +--ro root-address          inet:ipv4-address
      |  |        |     |     +--ro lsp-id                uint32
      |  |        |     |     +--ro recur-root-address    inet:ip-address
      |  |        |     |     +--ro recur-rd              route-distinguisher
      |  |        |     |     +--ro multipoint-type?      multipoint-type
      |  |        |     |     +--ro peer* [direction peer advertisement-type]
      |  |        |     |        +--ro direction             downstream-upstream
      |  |        |     |        +--ro peer                  leafref
      |  |        |     |        +--ro advertisement-type    advertised-received
      |  |        |     |        +--ro label?                mpls:mpls-label
      |  |        |     |        +--ro mbb-role?             enumeration
      |  |        |     |        +--ro mofrr-role?           enumeration
      |  |        |     +--ro opaque-type-src
      |  |        |     |  +--ro fec-label* [root-address source-address group-a
ddress rd recur-root-address recur-rd]
      |  |        |     |     +--ro root-address          inet:ipv4-address
      |  |        |     |     +--ro source-address        inet:ip-address
      |  |        |     |     +--ro group-address         inet:ip-address-no-zon
e
      |  |        |     |     +--ro rd                    route-distinguisher
      |  |        |     |     +--ro recur-root-address    inet:ip-address
      |  |        |     |     +--ro recur-rd              route-distinguisher
      |  |        |     |     +--ro multipoint-type?      multipoint-type
      |  |        |     |     +--ro peer* [direction peer advertisement-type]
      |  |        |     |        +--ro direction             downstream-upstream
      |  |        |     |        +--ro peer                  leafref
      |  |        |     |        +--ro advertisement-type    advertised-received
      |  |        |     |        +--ro label?                mpls:mpls-label
      |  |        |     |        +--ro mbb-role?             enumeration
      |  |        |     |        +--ro mofrr-role?           enumeration
      |  |        |     +--ro opaque-type-bidir
      |  |        |        +--ro fec-label* [root-address rp group-address rd re
cur-root-address recur-rd]
      |  |        |           +--ro root-address          inet:ipv4-address
      |  |        |           +--ro rp                    inet:ip-address
      |  |        |           +--ro group-address         inet:ip-address-no-zon
e
      |  |        |           +--ro rd                    route-distinguisher
      |  |        |           +--ro recur-root-address    inet:ip-address
      |  |        |           +--ro recur-rd              route-distinguisher
      |  |        |           +--ro multipoint-type?      multipoint-type
      |  |        |           +--ro peer* [direction peer advertisement-type]
      |  |        |              +--ro direction             downstream-upstream
      |  |        |              +--ro peer                  leafref
      |  |        |              +--ro advertisement-type    advertised-received
      |  |        |              +--ro label?                mpls:mpls-label
      |  |        |              +--ro mbb-role?             enumeration
      |  |        |              +--ro mofrr-role?           enumeration
```

```
      |  |        +--ro ipv6
      |  |           +--ro roots
      |  |           |  +--ro root* [root-address]
      |  |           |     +--ro root-address    inet:ipv6-address
      |  |           |     +--ro is-self?        boolean
      |  |           |     +--ro reachability* [address interface]
      |  |           |        +--ro address      inet:ipv6-address
      |  |           |        +--ro interface    mpls-interface-ref
      |  |           |        +--ro peer?        leafref
      |  |           +--ro bindings
      |  |              +--ro opaque-type-lspid
      |  |              |  +--ro fec-label* [root-address lsp-id recur-root-addr
ess recur-rd]
      |  |              |     +--ro root-address          inet:ipv6-address
      |  |              |     +--ro lsp-id                uint32
      |  |              |     +--ro recur-root-address    inet:ip-address
      |  |              |     +--ro recur-rd              route-distinguisher
      |  |              |     +--ro multipoint-type?      multipoint-type
      |  |              |     +--ro peer* [direction peer advertisement-type]
      |  |              |        +--ro direction             downstream-upstream
      |  |              |        +--ro peer                  leafref
      |  |              |        +--ro advertisement-type    advertised-received
      |  |              |        +--ro label?                mpls:mpls-label
      |  |              |        +--ro mbb-role?             enumeration
      |  |              |        +--ro mofrr-role?           enumeration
      |  |              +--ro opaque-type-src
      |  |              |  +--ro fec-label* [root-address source-address group-a
ddress rd recur-root-address recur-rd]
      |  |              |     +--ro root-address          inet:ipv6-address
      |  |              |     +--ro source-address        inet:ip-address
      |  |              |     +--ro group-address         inet:ip-address-no-zon
e
      |  |              |     +--ro rd                    route-distinguisher
      |  |              |     +--ro recur-root-address    inet:ip-address
      |  |              |     +--ro recur-rd              route-distinguisher
      |  |              |     +--ro multipoint-type?      multipoint-type
      |  |              |     +--ro peer* [direction peer advertisement-type]
      |  |              |        +--ro direction             downstream-upstream
      |  |              |        +--ro peer                  leafref
      |  |              |        +--ro advertisement-type    advertised-received
      |  |              |        +--ro label?                mpls:mpls-label
      |  |              |        +--ro mbb-role?             enumeration
      |  |              |        +--ro mofrr-role?           enumeration
      |  |              +--ro opaque-type-bidir
      |  |                 +--ro fec-label* [root-address rp group-address rd re
cur-root-address recur-rd]
      |  |                    +--ro root-address          inet:ipv6-address
      |  |                    +--ro rp                    inet:ip-address
      |  |                    +--ro group-address         inet:ip-address-no-zon
e
      |  |                    +--ro rd                    route-distinguisher
      |  |                    +--ro recur-root-address    inet:ip-address
      |  |                    +--ro recur-rd              route-distinguisher
```

```
|  |                    +--ro multipoint-type?      multipoint-type
|  |                    +--ro peer* [direction peer advertisement-type]
|  |                       +--ro direction             downstream-upstream
|  |                       +--ro peer                  leafref
|  |                       +--ro advertisement-type    advertised-received
|  |                       +--ro label?                mpls:mpls-label
|  |                       +--ro mbb-role?             enumeration
|  |                       +--ro mofrr-role?           enumeration
|  +--rw forwarding-nexthop {forwarding-nexthop-config}?
|     +--rw interfaces
|        +--rw interface* [interface]
|           +--rw address-family* [afi]
|              +--ro state
|                 +--ro mldp-disable?   boolean {mldp}?
|
+--rw peers
   +--rw peer* [lsr-id]
      +--ro state
         +--ro capability
         |   +--ro mldp {mldp}?
         |     +--ro p2mp
         |     |  +--ro enable?   boolean
         |     +--ro mp2mp
         |     |  +--ro enable?   boolean
         |     +--ro make-before-break
         |     |  +--ro enable?             boolean
         |     |  +--ro switchover-delay?   uint16
         |     |  +--ro timeout?            uint16
         |     +--ro hub-and-spoke {capability-mldp-hsmp}?
         |     |  +--ro enable?   boolean
         |     +--ro node-protection {capability-mldp-node-protection}?
         |        +--ro plr?           boolean
         |        +--ro merge-point
         |           +--ro enable?                                boolean
         |           +--ro targeted-session-teardown-delay?   uint16
         +--ro received-peer-state
            +--ro capability
               +--ro mldp {mldp}?
                  +--ro p2mp
                  |  +--ro enable?   boolean
                  +--ro mp2mp
                  |  +--ro enable?   boolean
                  +--ro make-before-break
                  |  +--ro enable?   boolean
                  +--ro hub-and-spoke
                  |  +--ro enable?   boolean
                  +--ro node-protection
                     +--ro plr?           boolean
                     +--ro merge-point?   boolean
```

Figure 15

4.3.1. Derived states

Following are main areas for which mLDP operational derived state is defined:

- Root
- Bindings (FEC-label)
- Capabilities

4.3.1.1. Root state

Root address is a fundamental construct for MP FEC bindings and LSPs. The root state provides information on all the known roots in a given address-family, and their information on the root reachability (as learnt from RIB). In case of multi-path reachability to a root, the selection of upstream path is done on per-LSP basis at the time of LSP setup. Similarly, when protection mechanisms like MBB or MoFRR are in place, the path designation as active/standby or primary/ backup is also done on per LSP basis. It is to be noted that a given root can be shared amongst multiple P2MP and/or MP2MP LSPs. Moreover, an LSP can be signaled to more than one root for RNR purposes.

The following diagram illustrates a root database on a branch/transit LSR:

```
        root 1.1.1.1:
            path1:
                RIB: GigEthernet 1/0, 12.1.0.2;
                LDP: peer 192.168.0.1:0
            path2:
                RIB: GigEthernet 2/0, 12.2.0.2;
                LDP: peer 192.168.0.3:0

        root 2.2.2.2:
            path1:
                RIB: 3.3.3.3;              (NOTE: This is a recursive path)
                LDP: peer 192.168.0.3:0    (NOTE: T-mLDP peer)

         root 9.9.9.9:
            . . . .
```

Figure 16

A root entry on a root LSR itself will be presented as follows:

```
          root 9.9.9.9:
              is-self
```

Figure 17

4.3.1.2. Bindings state

Binding state provides information on mLDP FEC-label bindings for both P2MP and MP2MP FEC types. Like LDP, the FEC-label binding derived state is presented in a FEC-centric view per address-family, and provides information on both inbound (received) and outbound (advertised) bindings. The FEC is presented as (root-address, opaque-type-data) and the direction (upstream or downstream) is picked with respect to root reachability. In case of MBB or/and MoFRR, the role of a given peer binding is also provided with respect to MBB (active or standby) or/and MoFRR (primary or backup).

This document covers following type of opaque values with their keys in the operational model of mLDP bindings:

Opaque Type	Key	RFC
Generic LSP Identifier	LSP Id	[RFC6388]
Transit IPv4 Source	Source, Group	[RFC6826]
Transit IPv6 Source	Source, Group	[RFC6826]
Transit IPv4 Bidir	RP, Group	[RFC6826]
Transit IPv6 Bidir	RP, Group	[RFC6826]
Transit VPNv4 Source	Source, Group, RD	[RFC7246]
Transit VPNv6 Source	Source, Group, RD	[RFC7246]
Transit VPNv4 Bidir	RP, Group, RD	[RFC7246]
Transit VPNv6 Bidir	RP, Group, RD	[RFC7246]
Recursive Opaque	Root	[RFC6512]
VPN-Recursive Opaque	Root, RD	[RFC6512]

Table 1: MP Opaque Types and keys

It is to be noted that there are three basic types (LSP Id, Source, and Bidir) and then there are variants (VPN, recursive, VPN-recursive) on top of these basic types.

Following captures high level tree hierarchy for mLDP bindings state:

```
+--rw mpls-ldp!
    +--rw mldp
      +--rw address-family* [afi]
         +--rw afi       address-family
         +--ro state
            +--ro ipv4 (or ipv6)
               +--ro bindings
                  +--ro opaque-type-xxx [root-address, type-specific-key]
                     +--ro root-address
                     +--ro ...
                     +--ro recur-root-address    inet:ipv4-address
                     +--ro recur-rd              route-distinguisher
                     +--ro multipoint-type?      multipoint-type
                     +--ro peer* [direction peer advertisement-type]
                        +--ro direction             downstream-upstream
                        +--ro peer                  leafref
                        +--ro advertisement-type    advertised-received
                        +--ro label?                mpls:mpls-label
                        +--ro mbb-role?             enumeration
                        +--ro mofrr-role?           enumeration
```

Figure 18

In the above tree, the type-specific-key varies with the base type as listed in earlier Table 1. For example, if the opaque type is Generic LSP Identifier, then the type-specific-key will be a uint32 value corresponding to the LSP. Please see the complete model for all other types.

Moreover, the binding tree defines only three types of sub-trees (i.e. lspid, src, and bidir) which is able to map the respective variants (vpn, recursive, and vpn-recusrive) accordingly. For example, the key for opaque-type-src is [R, S, G, rd, recur-R, recur-RD], where basic type will specify (R, S,G,-, -, -), VPN type will specify (R, S,G, rd, -, -), recursive type will specify [R, S,G, -, recur-R, -] and VPN-recursive type will specify [R, S,G, -, recur-R, recur-rd].

It is important to take note of the following:

- The address-family ipv4/ipv4 applies to "root" address in the mLDP binding tree. The other addresses (source, group, RP etc) do not have to be of the same address family type as the root.

- The "recur-root-address" field applies to Recursive opaque type, and (recur-root-address, recur-rd) fields applies to VPN-Recursive opaque types as defined in [RFC6512]

- In case of a recursive FEC, the address-family of the recur-root-address could be different than the address-family of the root address of original encapsulated MP FEC

The following diagram illustrates the FEC-label binding information structure for a P2MP (Transit IPv4 Source type) LSP on a branch/ transit LSR:

```
FEC (root 2.2.2.2, S=192.168.1.1, G=224.1.1.1):
  type: p2mp
  upstream:
    advertised:
      peer 192.168.0.1:0, label 16000 (local)
  downstream:
    received:
      peer 192.168.0.2:0, label 17000 (remote)
      peer 192.168.0.3:0, label 18000  (remote)
```

Figure 19

The following diagram illustrates the FEC-label binding information structure for a similar MP2MP LSP on a branch/transit LSR:

```
      FEC (root 2.2.2.2, RP=192.168.9.9, G=224.1.1.1):
        type: mp2mp
        upstream:
          advertised:
            peer 192.168.0.1:0, label 16000 (local)
          received:
            peer 192.168.0.1:0, label 17000 (remote)
        downstream:
          advertised:
            peer 192.168.0.2:0, label 16001 (local), MBB role=active
            peer 192.168.0.3:0, label 16002 (local), MBB role=standby
          received:
            peer 192.168.0.2:0, label 17001 (remote)
            peer 192.168.0.3:0, label 18001 (remote)
```

Figure 20

4.3.1.3. Capabilities state

Like LDP, mLDP capabilities state comprise two types of information - global information and per-peer information.

4.4. Notifications

mLDP notification module consists of notification related to changes in the operational state of an mLDP FEC. Following is a simplified graphical representation of the data model for mLDP notifications:

```
notifications:
   +---n mpls-mldp-fec-event
      +--ro event-type?       oper-status-event-type
      +--ro tree-type?        multipoint-type
      +--ro root?             inet:ip-address
      +--ro (lsp-key-type)?
         +--:(lsp-id-based)
         |  +--ro lsp-id?           uint16
         +--:(source-group-based)
            +--ro source-address?   inet:ip-address
            +--ro group-address?    inet:ip-address
```

Figure 21

4.5. Actions

Currently, no RPCs/actions are defined for mLDP.

5. Open Items

Following is a list of open items that are to be discussed and addressed in future revisions of this document:

- Close on augmentation off "mpls" list in "ietf-mpls" defined in [I-D.saad-mpls-base-yang]
- Align operational state modeling with other routing procols and [I-D.openconfig-netmod-opstate]
- Complete the section on Protocol-centric implementations and all-vrfs
- Specify default values for configuration parameters
- Revisit and cut down on the scope of the document and number of features it is trying to cover
- Split the model into a base and extended items
- Add statistics for mLDP root LSPs and bindings
- Extend the "Configured Leaf LSPs" for various type of opaque-types
- Extend mLDP notifications for other types of opaque values as well
- Close on single vs separate document for mLDP Yang

6. YANG Specification

Following are actual YANG definition for LDP and mLDP constructs defined earlier in the document.

```
 <CODE BEGINS> file "ietf-mpls-ldp@2016-07-08.yang" -->

 module ietf-mpls-ldp {
   namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-ldp";
   // replace with IANA namespace when assigned
   prefix ldp;

   import ietf-inet-types {
```

```
    prefix "inet";
  }

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

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-ip {
    prefix "ip";
  }

  import ietf-routing {
    prefix "rt";
  }

  import ietf-mpls {
    prefix "mpls";
  }

  organization
    "IETF MPLS Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/teas/>
     WG List:  <mailto:teas@ietf.org>

     WG Chair: Loa Andersson
               <mailto:loa@pi.nu>

     WG Chair: Ross Callon
               <mailto:rcallon@juniper.net>

     WG Chair: George Swallow
               <mailto:swallow.ietf@gmail.com>

     Editor:   Kamran Raza
               <mailto:skraza@cisco.com>

     Editor:   Rajiv Asati
               <mailto:rajiva@cisco.com>

     Editor:   Xufeng Liu
               <mailto:xliu@kuatrotech.com>

     Editor:   Santosh Esale
```

```
                <mailto:sesale@juniper.net>

       Editor:  Xia Chen
                <mailto:jescia.chenxia@huawei.com>

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

   description
     "This YANG module defines the essential components for the
      management of Multi-Protocol Label Switching (MPLS) Label
      Distribution Protocol (LDP) and Multipoint LDP (mLDP).";

   revision 2016-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: YANG Data Model for MPLS LDP and mLDP.";
   }

   /*
    * Features
    */

   feature admin-down-config {
     description
       "This feature indicates that the system allows to configure
        administrative down on a VRF instance and a peer.";
   }

   feature all-af-policy-config {
     description
       "This feature indicates that the system allows to configure
        policies that are applied to all address families.";
   }

   feature capability-end-of-lib {
     description
       "This feature indicates that the system allows to configure
        LDP end-of-lib capability.";
   }

   feature capability-mldp-hsmp {
     description
       "This feature indicates that the system allows to configure
        mLDP hub-and-spoke-multipoint capability.";
   }
```

```
   feature capability-mldp-node-protection {
     description
       "This feature indicates that the system allows to configure
        mLDP node-protection capability.";
   }

   feature capability-typed-wildcard-fec {
     description
       "This feature indicates that the system allows to configure
        LDP typed-wildcard-fec capability.";
   }

   feature capability-upstream-label-assignment {
     description
       "This feature indicates that the system allows to configure
        LDP upstream label assignment capability.";
   }

   feature forwarding-nexthop-config {
     description
       "This feature indicates that the system allows to configure
        forwarding nexthop on interfaces.";
   }

   feature global-session-authentication {
     description
       "This feature indicates that the system allows to configure
        authentication at global level.";
   }

   feature graceful-restart-helper-mode {
     description
       "This feature indicates that the system supports graceful
        restart helper mode.";
   }

   feature mldp {
     description
       "This feature indicates that the system supports Multicast
        LDP (mLDP).";
   }

   feature mldp-mofrr {
     description
       "This feature indicates that the system supports mLDP
        Multicast only FRR (MoFRR).";
   }
```

```
   feature per-interface-timer-config {
     description
       "This feature indicates that the system allows to configure
        interface hello timers at the per-interface level.";
   }

   feature per-peer-graceful-restart-config {
     description
       "This feature indicates that the system allows to configure
        graceful restart at the per-peer level.";
   }

   feature per-peer-session-attributes-config {
     description
       "This feature indicates that the system allows to configure
        session attributes at the per-peer level.";
   }

   feature policy-extended-discovery-config {
     description
       "This feature indicates that the system allows to configure
        policies to control the acceptance of extended neighbor
        discovery hello messages.";
   }

   feature policy-label-assignment-config {
     description
       "This feature indicates that the system allows to configure
        policies to assign labels according to certain prefixes.";
   }

   feature policy-ordered-label-config {
     description
       "This feature indicates that the system allows to configure
        ordered label policies.";
   }

   feature session-downstream-on-demand-config {
     description
       "This feature indicates that the system allows to configure
        session downstream-on-demand";
   }

   /*
    * Typedefs
    */
   typedef ldp-address-family {
     type identityref {
```

```
        base rt:address-family;
      }
      description
        "LDP address family type.";
    }

    typedef duration32-inf {
      type union {
        type uint32;
        type enumeration {
          enum "infinite" {
            description "The duration is infinite.";
          }
        }
      }
      units seconds;
      description
        "Duration represented as 32 bit seconds with infinite.";
    }

    typedef advertised-received {
      type enumeration {
        enum advertised {
          description "Advertised information.";
        }
        enum received {
          description "Received information.";
        }
      }
      description
        "Received or advertised.";
    }

    typedef downstream-upstream {
      type enumeration {
        enum downstream {
          description "Downstream information.";
        }
        enum upstream {
          description "Upstream information.";
        }
      }
      description
        "Received or advertised.";
    }

    typedef label-adv-mode {
      type enumeration {
```

```
        enum downstream-unsolicited {
          description "Downstream Unsolicited.";
        }
        enum downstream-on-demand {
          description "Downstream on Demand.";
        }
      }
      description
        "Label Advertisement Mode.";
    }

    typedef mpls-interface-ref {
      type leafref {
        path "/rt:routing/mpls:mpls/mpls:interface/mpls:name";
      }
      description
        "This type is used by data models that need to reference
         mpls interfaces.";
    }

    typedef multipoint-type {
      type enumeration {
        enum p2mp {
          description "Point to multipoint.";
        }
        enum mp2mp {
          description "Multipoint to multipoint.";
        }
      }
      description
        "p2mp or mp2mp.";
    }

    typedef neighbor-list-ref {
      type string;
      description
        "A type for a reference to a neighbor list.";
    }

    typedef peer-list-ref {
      type string;
      description
        "A type for a reference to a peer list.";
    }

    typedef prefix-list-ref {
      type string;
      description
```

```
      "A type for a reference to a prefix list.";
  }

  typedef oper-status-event-type {
    type enumeration {
      enum up {
        value 1;
        description
          "Operational status changed to up.";
      }
      enum down {
        value 2;
        description
          "Operational status changed to down.";
      }
    }
    description "Operational status event type for notifications.";
  }

  typedef route-distinguisher {
    type string {
    }
    description
      "Type definition for route distinguisher.";
    reference
      "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
  }

  /*
   * Identities
   */
  identity adjacency-flag-base {
    description "Base type for adjacency flags.";
  }

  identity adjacency-flag-active {
    base "adjacency-flag-base";
    description
      "This adjacency is configured and actively created.";
  }

  identity adjacency-flag-passive {
    base "adjacency-flag-base";
    description
      "This adjacency is not configured and passively accepted.";
  }

  /*
```

```
    * Groupings
    */

   grouping adjacency-state-attributes {
     description
       "Adjacency state attributes.";

     leaf-list flag {
       type identityref {
         base "adjacency-flag-base";
       }
       description "Adjacency flags.";
     }
     container hello-holdtime {
       description "Hello holdtime state.";
       leaf adjacent {
         type uint16;
         units seconds;
         description "Peer holdtime.";
       }
       leaf negotiated {
         type uint16;
         units seconds;
         description "Negotiated holdtime.";
       }
       leaf remaining {
         type uint16;
         units seconds;
         description "Remaining holdtime.";
       }
     }

     leaf next-hello {
       type uint16;
       units seconds;
       description "Time to send the next hello message.";
     }

     container statistics {
       description
         "Statistics objects.";

       leaf discontinuity-time {
         type yang:date-and-time;
         mandatory true;
         description
           "The time on the most recent occasion at which any one or
            more of this interface's counters suffered a
```

```
            discontinuity.  If no such discontinuities have occurred
            since the last re-initialization of the local management
            subsystem, then this node contains the time the local
            management subsystem re-initialized itself.";
        }

        leaf hello-received {
          type yang:counter64;
          description
            "The number of hello messages received.";
        }
        leaf hello-dropped {
          type yang:counter64;
          description
            "The number of hello messages received.";
        }
      } // statistics
    } // adjacency-state-attributes

    grouping basic-discovery-timers {
      description
        "Basic discovery timer attributes.";
      leaf hello-holdtime {
        type uint16 {
          range 15..3600;
        }
        units seconds;
        description
          "The time interval for which a LDP link Hello adjacency
           is maintained in the absence of link Hello messages from
           the LDP neighbor";
      }
      leaf hello-interval {
        type uint16 {
          range 5..1200;
        }
        units seconds;
        description
          "The interval between consecutive LDP link Hello messages
           used in basic LDP discovery";
      }
    } // basic-discovery-timers

    grouping binding-address-state-attributes {
      description
        "Address binding attributes";
      leaf advertisement-type {
        type advertised-received;
```

```
      description
        "Received or advertised.";
    }
    leaf peer {
      type leafref {
        path "../../../../../../../peers/peer/lsr-id";
      }
      must "../advertisement-type = 'received'" {
        description
          "Applicable for received address.";
      }
      description
        "LDP peer from which this address is received.";
    } // peer
  } // binding-address-state-attributes

  grouping binding-label-state-attributes {
    description
      "Label binding attributes";
    list peer {
      key "peer advertisement-type";
      description
        "List of advertised and received peers.";
      leaf peer {
        type leafref {
          path "../../../../../../../../peers/peer/lsr-id";
        }
        description
          "LDP peer from which this binding is received,
           or to which this binding is advertised.";
      }
      leaf advertisement-type {
        type advertised-received;
        description
          "Received or advertised.";
      }
      leaf label {
        type mpls:mpls-label;
        description
          "Advertised (outbound) or received (inbound)
           label.";
      }
      leaf used-in-forwarding {
        type boolean;
        description
          "'true' if the lable is used in forwarding.";
      }
    } // peer
```

```
   } // binding-label-state-attributes

   grouping extended-discovery-policy-attributes {
     description
       "LDP policy to control the acceptance of extended neighbor
        discovery hello messages.";
     container hello-accept {
       if-feature policy-extended-discovery-config;
       description
         "Extended discovery acceptance policies.";

       leaf enable {
         type boolean;
         description
           "'true' to accept; 'false' to deny.";
       }
       leaf neighbor-list {
         type neighbor-list-ref;
         description

           "The name of a peer ACL.";
       }
     } // hello-accept
   } // extended-discovery-policy-attributes

   grouping extended-discovery-timers {
     description
       "Extended discovery timer attributes.";
     leaf hello-holdtime {
       type uint16 {
         range 15..3600;
       }
       units seconds;
       description
         "The time interval for which LDP targeted Hello adjacency

          is maintained in the absence of targeted Hello messages
          from an LDP neighbor.";
     }
     leaf hello-interval {
       type uint16 {
         range 5..3600;
       }
       units seconds;
       description
         "The interval between consecutive LDP targeted Hello
          messages used in extended LDP discovery.";
     }
```

```
   } // extended-discovery-timers

   grouping global-attributes {
     description "Configuration attributes at global level.";

     uses instance-attributes;
   } // global-attributes

   grouping graceful-restart-attributes {
     description
       "Graceful restart configuration attributes.";
     container graceful-restart {
       description
         "Attributes for graceful restart.";
       leaf enable {
         type boolean;
         description
           "Enable or disable graceful restart.";
       }
       leaf helper-enable {
         if-feature graceful-restart-helper-mode;
         type boolean;
         description
           "Enable or disable graceful restart helper mode.";
       }
       leaf reconnect-time {
         type uint16 {
           range 10..1800;
         }
         units seconds;
         description
           "Specifies the time interval that the remote LDP peer
            must wait for the local LDP peer to reconnect after the
            remote peer detects the LDP communication failure.";
       }
       leaf recovery-time {
         type uint16 {
           range 30..3600;
         }
         units seconds;
         description
           "Specifies the time interval, in seconds, that the remote
            LDP peer preserves its MPLS forwarding state after
            receiving the Initialization message from the restarted
            local LDP peer.";
       }
       leaf forwarding-holdtime {
         type uint16 {
```

```
          range 30..3600;
        }
        units seconds;
        description
          "Specifies the time interval, in seconds, before the
           termination of the recovery phase.";
      }
    } // graceful-restart
  } // graceful-restart-attributes

  grouping graceful-restart-attributes-per-peer {
    description
      "Per peer graceful restart configuration attributes.";
    container graceful-restart {
      description
        "Attributes for graceful restart.";
      leaf enable {
        type boolean;
        description
          "Enable or disable graceful restart.";
      }
      leaf reconnect-time {
        type uint16 {
          range 10..1800;
        }
        units seconds;
        description
          "Specifies the time interval that the remote LDP peer
           must wait for the local LDP peer to reconnect after the
           remote peer detects the LDP communication failure.";
      }
      leaf recovery-time {
        type uint16 {
          range 30..3600;
        }
        units seconds;
        description
          "Specifies the time interval, in seconds, that the remote
           LDP peer preserves its MPLS forwarding state after
           receiving the Initialization message from the restarted
           local LDP peer.";
      }
    } // graceful-restart
  } // graceful-restart-attributes-per-peer

  grouping instance-attributes {
    description "Configuration attributes at instance level.";
```

```
    container capability {
      description "Configure capability.";
      container end-of-lib {
        if-feature capability-end-of-lib;
        description
          "Configure end-of-lib capability.";
        leaf enable {
          type boolean;
          description
            "Enable end-of-lib capability.";
        }
      }
      container typed-wildcard-fec {
        if-feature capability-typed-wildcard-fec;
        description
          "Configure typed-wildcard-fec capability.";
        leaf enable {
          type boolean;
          description
            "Enable typed-wildcard-fec capability.";
        }
      }
      container upstream-label-assignment {
        if-feature capability-upstream-label-assignment;
        description
          "Configure upstream label assignment capability.";
        leaf enable {
          type boolean;
          description
            "Enable upstream label assignment.";
        }
      }
      container mldp {
        if-feature mldp;

        description
          "Multipoint capabilities.";
        uses mldp-capabilities;
      }
    } // capability

    uses graceful-restart-attributes;

    leaf igp-synchronization-delay {
      type uint16 {
        range 3..60;
      }
      units seconds;
```

```
      description
        "Sets the interval that the LDP waits before notifying the
         Interior Gateway Protocol (IGP) that label exchange is
         completed so that IGP can start advertising the normal
         metric for the link.";
    }
    leaf lsr-id {
      type yang:dotted-quad;
      description "Router ID.";
    }
  } // instance-attributes

  grouping ldp-adjacency-ref {
    description
      "An absolute reference to an LDP adjacency.";
    choice hello-adjacency-type {
      description
        "Interface or targeted adjacency.";
      case targeted {
        container targeted {
          description "Targeted adjacency.";
          leaf target-address {
            type inet:ip-address;
            description
              "The target address.";
          }
        } // targeted
      }
      case link {
        container link {
          description "Link adjacency.";
          leaf next-hop-interface {
            type mpls-interface-ref;
            description
              "Interface connecting to next-hop.";
          }
          leaf next-hop-address {
            type inet:ip-address;
            must "../next-hop-interface" {
              description
                "Applicable when interface is specified.";

            }
            description
              "IP address of next-hop.";
          }
        } // link
      }
```

```
    }
  } // ldp-adjacency-ref

  grouping ldp-fec-event {
    description
      "A LDP FEC event.";
    leaf prefix {
      type inet:ip-prefix;
      description
        "FEC.";
    }
  } // ldp-fec-event

  grouping ldp-peer-ref {
    description
      "An absolute reference to an LDP peer.";
    leaf peer-ref {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/mpls-ldp/"
          + "peers/peer/lsr-id";
      }
      description
        "Reference to an LDP peer.";
    }
  } // ldp-peer-ref

  grouping mldp-capabilities {
    description
      "mLDP capabilities.";
    container p2mp {
      description
        "Configure point-to-multipoint capability.";
      leaf enable {
        type boolean;
        description
          "Enable point-to-multipoint.";
      }
    }
    container mp2mp {
      description
        "Configure multipoint-to-multipoint capability.";
      leaf enable {
        type boolean;
        description
          "Enable multipoint-to-multipoint.";
      }
    }
    container make-before-break {
```

```
        description
          "Configure make-before-break capability.";
        leaf enable {
          type boolean;
          description
            "Enable make-before-break.";
        }
        leaf switchover-delay {
          type uint16;
          units seconds;
          description
            "Switchover delay in seconds.";
        }
        leaf timeout {
          type uint16;
          units seconds;
          description
            "Timeout in seconds.";
        }
      }
      container hub-and-spoke {
        if-feature capability-mldp-hsmp;
        description
          "Configure hub-and-spoke-multipoint capability.";
        reference
          "RFC7140: LDP Extensions for Hub and Spoke Multipoint
           Label Switched Path";
        leaf enable {
          type boolean;
          description
            "Enable hub-and-spoke-multipoint.";
        }
      }
      container node-protection {
        if-feature capability-mldp-node-protection;
        description
          "Configure node-protection capability.";
        reference
          "RFC7715: mLDP Node Protection.";
        leaf plr {
          type boolean;
          description
            "Point of Local Repair capable for MP LSP node
             protection.";
        }
        container merge-point {
          description
            "Merge Point capable for MP LSP node protection.";
```

```
        leaf enable {
          type boolean;
          description
            "Enable merge point capability.";
        }
        leaf targeted-session-teardown-delay {
          type uint16;
          units seconds;
          description
            "Targeted session teardown delay.";
        }
      } // merge-point
    }
  } // mldp-capabilities

  grouping  mldp-configured-lsp-roots {
    description
      "mLDP roots containers.";

    container roots-ipv4 {

      when "../../../af = 'ipv4'" {
        description
          "Only for IPv4.";
      }
      description
        "Configured IPv4 multicast LSPs.";
      list root {
        key "root-address";
        description
          "List of roots for configured multicast LSPs.";

        leaf root-address {
          type inet:ipv4-address;
          description
            "Root address.";
        }

        list lsp {
          must "(lsp-id = 0 and source-address != '0.0.0.0' and "
            + "group-address != '0.0.0.0') or "
            + "(lsp-id != 0 and source-address = '0.0.0.0' and "
            + "group-address = '0.0.0.0')" {
            description
              "A LSP can be identified by either <lsp-id> or
               <source-address, group-address>.";
          }
          key "lsp-id source-address group-address";
```

```
          description
            "List of LSPs.";
          leaf lsp-id {
            type uint16;
            description "ID to identify the LSP.";
          }
          leaf source-address {
            type inet:ipv4-address;
            description
              "Source address.";
          }
          leaf group-address {
            type inet:ipv4-address-no-zone;
            description
              "Group address.";
          }
        } // list lsp
      } // list root
    } // roots-ipv4

    container roots-ipv6 {

      when "../../../af = 'ipv6'" {
        description
          "Only for IPv6.";
      }
      description
        "Configured IPv6 multicast LSPs.";

      list root {
        key "root-address";
        description
          "List of roots for configured multicast LSPs.";

        leaf root-address {
          type inet:ipv6-address;
          description
            "Root address.";
        }

        list lsp {
          must "(lsp-id = 0 and source-address != '::' and "
            + "group-address != '::') or "
            + "(lsp-id != 0 and source-address = '::' and "
            + "group-address = '::')" {
            description
              "A LSP can be identified by either <lsp-id> or
               <source-address, group-address>.";
```

```
          }
          key "lsp-id source-address group-address";
          description
            "List of LSPs.";
          leaf lsp-id {
            type uint16;
            description "ID to identify the LSP.";
          }
          leaf source-address {
            type inet:ipv6-address;
            description
              "Source address.";
          }
          leaf group-address {
            type inet:ipv6-address-no-zone;
            description
              "Group address.";
          }
        } // list lsp
      } // list root
    } // roots-ipv6
  } // mldp-configured-lsp-roots

  grouping mldp-fec-event {
    description
      "A mLDP FEC event.";
    leaf tree-type {
      type multipoint-type;
      description
        "p2mp or mp2mp.";
    }
    leaf root {
      type inet:ip-address;
      description
        "Root address.";
    }
    choice lsp-key-type {
      description
        "LSP ID based or source-group based .";
      case lsp-id-based {
        leaf lsp-id {
          type uint16;
          description
            "ID to identify the LSP.";
        }
      }
      case source-group-based {
        leaf source-address {
```

```
          type inet:ip-address;
          description

            "LSP source address.";
        }
        leaf group-address {
          type inet:ip-address;
          description
            "Multicast group address.";
        }
      } // case source-group-based
    }
  } // mldp-fec-event

  grouping  mldp-binding-label-state-attributes {
    description
      "mLDP label binding attributes.";

    leaf multipoint-type {
      type multipoint-type;
      description
        "The type of mutipoint, p2mp or mp2mp.";
    }
    list peer {
      key "direction peer advertisement-type";
      description
        "List of advertised and received peers.";
      leaf direction {
        type downstream-upstream;
        description
          "Downstream or upstream.";
      }
      leaf peer {
        type leafref {
          path
            "../../../../../../../../../../peers/peer/lsr-id";
        }
        description
          "LDP peer from which this binding is received,
           or to which this binding is advertised.";
      }
      leaf advertisement-type {
        type advertised-received;
        description
          "Advertised or received.";
      }
      leaf label {
        type mpls:mpls-label;
```

```
          description
            "Advertised (outbound) or received (inbound) label.";
        }
        leaf mbb-role {
          when "../direction = 'upstream'" {
            description
              "For upstream.";
          }
          type enumeration {
            enum none {
              description "MBB is not enabled.";
            }
            enum active {
              description "This LSP is active.";
            }
            enum inactive {
              description "This LSP is inactive.";
            }
          }
          description
            "The MBB status of this LSP.";
        }
        leaf mofrr-role {
          when "../direction = 'upstream'" {
            description
              "For upstream.";
          }
          type enumeration {
            enum none {
              description "MOFRR is not enabled.";
            }
            enum primary {
              description "This LSP is primary.";
            }
            enum backup {
              description "This LSP is backup.";
            }
          }
          description
            "The MOFRR status of this LSP.";
        }
      } // peer
    } // mldp-binding-label-state-attributes

    grouping peer-af-policy-container {
      description
        "LDP policy attribute container under peer address-family.";
      container label-policy {
```

```
      description
        "Label policy attributes.";
      container advertise {
        description
          "Label advertising policies.";
        leaf prefix-list {
          type prefix-list-ref;
            description
              "Applies the prefix list to outgoing label
               advertisements.";
        }
      }
      container accept {
        description
          "Label advertisement acceptance policies.";
        leaf prefix-list {
          type prefix-list-ref;
          description
            "Applies the prefix list to incoming label
             advertisements.";
        }
      } // accept
    } // label-policy
  } // peer-af-policy-container

  grouping peer-attributes {
    description "Peer configuration attributes.";

    leaf session-ka-holdtime {
      type uint16 {
        range 45..3600;
      }
      units seconds;
      description
        "The time interval after which an inactive LDP session
         terminates and the corresponding TCP session closes.
         Inactivity is defined as not receiving LDP packets from the
         peer.";
    }
    leaf session-ka-interval {
      type uint16 {
        range 15..1200;
      }
      units seconds;
      description
        "The interval between successive transmissions of keepalive
         packets. Keepalive packets are only sent in the absence of
         other LDP packets transmitted over the LDP session.";
```

```
    }
  } // peer-attributes

  grouping peer-authentication {
    description
      "Peer authentication attributes.";
    leaf session-authentication-md5-password {
      type string {
        length "1..80";
      }
      description
        "Assigns an encrypted MD5 password to an LDP
         peer";
    } // md5-password
  } // peer-authentication

  grouping peer-state-derived {
    description "Peer derived state attributes.";

    container label-advertisement-mode {
      description "Label advertisement mode state.";
      leaf local {
        type label-adv-mode;
        description
          "Local Label Advertisement Mode.";
      }
      leaf peer {
        type label-adv-mode;
        description
          "Peer Label Advertisement Mode.";
      }
      leaf negotiated {
        type label-adv-mode;
        description
          "Negotiated Label Advertisement Mode.";
      }
    }
    leaf next-keep-alive {
      type uint16;
      units seconds;
      description "Time to send the next KeepAlive message.";
    }

    leaf peer-ldp-id {
      type yang:dotted-quad;
      description "Peer LDP ID.";
    }
```

```
     container received-peer-state {
       description "Peer features.";

       uses graceful-restart-attributes-per-peer;

       container capability {
         description "Configure capability.";
         container end-of-lib {
           description
             "Configure end-of-lib capability.";
           leaf enable {
             type boolean;
             description
               "Enable end-of-lib capability.";
           }
         }
         container typed-wildcard-fec {
           description
             "Configure typed-wildcard-fec capability.";
           leaf enable {
             type boolean;
             description
               "Enable typed-wildcard-fec capability.";
           }
         }
         container upstream-label-assignment {
           description
             "Configure upstream label assignment capability.";
           leaf enable {
             type boolean;
             description
               "Enable upstream label assignment.";
           }
         }
         container mldp {
           if-feature mldp;
           description
             "Multipoint capabilities.";

           container p2mp {
             description
               "Configure point-to-multipoint capability.";
             leaf enable {
               type boolean;
               description
                 "Enable point-to-multipoint.";
             }
           }
```

```
            container mp2mp {
              description
                "Configure multipoint-to-multipoint capability.";
              leaf enable {
                type boolean;
                description
                  "Enable multipoint-to-multipoint.";
              }
            }
            container make-before-break {
              description
                "Configure make-before-break capability.";
              leaf enable {
                type boolean;
                description
                  "Enable make-before-break.";
              }
            }
            container hub-and-spoke {
              description
                "Configure hub-and-spoke-multipoint capability.";
              reference
                "RFC7140: LDP Extensions for Hub and Spoke Multipoint
                 Label Switched Path";
              leaf enable {
                type boolean;
                description
                  "Enable hub-and-spoke-multipoint.";
              }
            }
            container node-protection {
              description
                "Configure node-protection capability.";
              reference
                "RFC7715: mLDP Node Protection.";
              leaf plr {
                type boolean;
                description
                  "Point of Local Repair capable for MP LSP node
                   protection.";
              }
              leaf merge-point {
                type boolean;
                description
                  "Merge Point capable for MP LSP node protection.";
              } // merge-point
            } // node-protection
          } // mldp
```

```
        } // capability
      } // received-peer-state

      container session-holdtime {
        description "Session holdtime state.";
        leaf peer {
          type uint16;
          units seconds;
          description "Peer holdtime.";
        }
        leaf negotiated {
          type uint16;
          units seconds;
          description "Negotiated holdtime.";
        }
        leaf remaining {
          type uint16;
          units seconds;
          description "Remaining holdtime.";
        }
      } // session-holdtime

      leaf session-state {
        type enumeration {
          enum non-existent {
            description "NON EXISTENT state. Transport disconnected.";
          }
          enum initialized {
            description "INITIALIZED state.";
          }
          enum openrec {
            description "OPENREC state.";
          }
          enum opensent {
            description "OPENSENT state.";
          }
          enum operational {
            description "OPERATIONAL state.";
          }
        }
        description
          "Representing the operational status.";
      }

      container tcp-connection {
        description "TCP connection state.";
        leaf local-address {
          type inet:ip-address;
```

```
        description "Local address.";
      }
      leaf local-port {
        type inet:port-number;
        description "Local port.";
      }
      leaf remote-address {
        type inet:ip-address;
        description "Remote address.";
      }
      leaf remote-port {
        type inet:port-number;
        description "Remote port.";
      }
    } // tcp-connection

    leaf up-time {
      type string;
      description "Up time. The interval format in ISO 8601.";
    }

    container statistics {
      description
        "Statistics objects.";

      leaf discontinuity-time {
        type yang:date-and-time;
        mandatory true;
        description
          "The time on the most recent occasion at which any one or
           more of this interface's counters suffered a
           discontinuity.  If no such discontinuities have occurred
           since the last re-initialization of the local management
           subsystem, then this node contains the time the local
           management subsystem re-initialized itself.";
      }

      container received {
        description "Inbound statistics.";
        uses statistics-peer-received-sent;
      }
      container sent {
        description "Outbound statistics.";
        uses statistics-peer-received-sent;
      }

      leaf total-addresses {
        type uint32;
```

```
        description
          "The number of learned addresses.";
      }
      leaf total-labels {
        type uint32;
        description
          "The number of learned labels.";
      }
      leaf total-fec-label-bindings {
        type uint32;
        description
          "The number of learned label-address bindings.";
      }
    } // statistics
  } // peer-state-derived

  grouping policy-container {
    description
      "LDP policy attributes.";
    container label-policy {
      description
        "Label policy attributes.";
      container independent-mode {
        description
          "Independent label policy attributes.";
        container assign {

          if-feature policy-label-assignment-config;
          description
            "Label assignment policies";
          choice prefix-option {
            description
              "Use either prefix-list or host-routes-only.";
            case prefix-list {
              leaf prefix-list {
                type prefix-list-ref;
                description
                  "Assign labels according to certain prefixes.";
              }
            }
            case host-routes-only {
              leaf host-routes-only {
                type boolean;
                description
                  "'true' to apply host routes only.";
              }
            }
          } // prefix-option
```

```
        }
        container advertise {
          description
            "Label advertising policies.";
          container explicit-null {
            description
              "Enables an egress router to advertise an
               explicit null label (value 0) in place of an
               implicit null label (value 3) to the
               penultimate hop router.";
            leaf enable {
              type boolean;
              description
                "'true' to enable explicit null.";
            }
            leaf prefix-list {
              type prefix-list-ref;
              description
                "Prefix list name. Applies the filters in the
                 specified prefix list to label
                 advertisements.
                 If the prefix list is not specified, explicit
                 null label advertisement is enabled for all
                 directly connected prefixes.";
            }
          }
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to outgoing label
               advertisements.";

          }
        }
        container accept {
          description
            "Label advertisement acceptance policies.";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to incoming label
               advertisements.";
          }
        }
      } // independent-mode
      container ordered-mode {
        if-feature policy-ordered-label-config;
        description
```

```
          "Ordered label policy attributes.";
        container egress-lsr {
          description
            "Egress LSR label assignment policies";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Assign labels according to certain prefixes.";
          }
        }
        container advertise {
          description
            "Label advertising policies.";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to outgoing label
               advertisements.";
          }
        }
        container accept {
          description
            "Label advertisement acceptance policies.";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to incoming label
               advertisements.";
          }
        }
      } // ordered-mode
    } // label-policy
  } // policy-container

  grouping statistics-peer-received-sent {
    description
      "Inbound and outbound statistic counters.";
    leaf total-octets {
      type yang:counter64;
      description
        "The total number of octets sent or received.";
    }
    leaf total-messages {
      type yang:counter64;
      description
        "The number of messages sent or received.";
    }
    leaf address {
```

```
        type yang:counter64;
        description
          "The number of address messages sent or received.";
      }
      leaf address-withdraw {
        type yang:counter64;
        description
          "The number of address-withdraw messages sent or received.";
      }
      leaf initialization {
        type yang:counter64;
        description
          "The number of initialization messages sent or received.";
      }
      leaf keepalive {
        type yang:counter64;
        description
          "The number of keepalive messages sent or received.";
      }
      leaf label-abort-request {
        type yang:counter64;
        description
          "The number of label-abort-request messages sent or
           received.";
      }
      leaf label-mapping {
        type yang:counter64;
        description
          "The number of label-mapping messages sent or received.";
      }
      leaf label-release {
        type yang:counter64;
        description
          "The number of label-release messages sent or received.";
      }
      leaf label-request {
        type yang:counter64;
        description
          "The number of label-request messages sent or received.";
      }
      leaf label-withdraw {
        type yang:counter64;
        description
          "The number of label-withdraw messages sent or received.";
      }
      leaf notification {
        type yang:counter64;
        description
```

```
          "The number of messages sent or received.";
      }
    } // statistics-peer-received-sent

    /*
     * Configuration data nodes
     */

    augment "/rt:routing/rt:control-plane-protocols" {
      description "LDP augmentation.";

      container mpls-ldp {
        presence "Container for LDP protocol.";
        description
          "Container for LDP protocol.";

        container global {
          description
            "Global attributes for LDP.";
          container config {
            description
              "Configuration data.";
            uses global-attributes;
          }
          container state {
            config false;
            description
              "Operational state data.";
            uses global-attributes;
          }

          container mldp {
            if-feature mldp;
            description
              "mLDP attributes at per instance level. Defining
               attributes here does not enable any MP capabilities.
               MP capabilities need to be explicitly enabled under
               container capability.";

            container config {
              description
                "Configuration data.";
              leaf enable {
                type boolean;
                description
                  "Enable mLDP.";
              }
            }
```

```
            container state {
              config false;
              description

                "Operational state data.";
              leaf enable {
                type boolean;
                description
                  "Enable mLDP.";
              }
            }

            list address-family {
              key "afi";
              description
                "Per-af params.";
              leaf afi {
                type ldp-address-family;
                description
                  "Address family type value.";
              }

              container config {
                description
                  "Configuration data.";
                container multicast-only-frr {
                  if-feature mldp-mofrr;
                  description
                    "Multicast only FRR (MoFRR) policy.";
                  leaf prefix-list {
                    type prefix-list-ref;
                    description
                      "Enables MoFRR for the specified access list.";
                  }
                } // multicast-only-frr
                container recursive-fec {
                  description
                    "Recursive FEC policy.";
                  leaf prefix-list {
                    type prefix-list-ref;
                    description
                      "Enables recursive FEC for the specified access
                       list.";
                  }
                } // recursive-for
              }
              container state {
                config false;
```

```
                description
                  "Operational state data.";
                container multicast-only-frr {
                  if-feature mldp-mofrr;

                  description
                    "Multicast only FRR (MoFRR) policy.";
                  leaf prefix-list {
                    type prefix-list-ref;
                    description
                      "Enables MoFRR for the specified access list.";
                  }
                } // multicast-only-frr
                container recursive-fec {
                  description
                    "Recursive FEC policy.";
                  leaf prefix-list {
                    type prefix-list-ref;
                    description
                      "Enables recursive FEC for the specified access
                       list.";
                  }
                } // recursive-fec

                container ipv4 {
                  when "../../afi = 'ipv4'" {
                    description
                      "Only for IPv4.";
                  }
                  description
                    "IPv4 state information.";
                  container roots {
                    description
                      "IPv4 multicast LSP roots.";
                    list root {
                      key "root-address";
                      description
                        "List of roots for configured multicast LSPs.";

                      leaf root-address {
                        type inet:ipv4-address;
                        description
                          "Root address.";
                      }

                      leaf is-self {
                        type boolean;
                        description
```

```
                    "This is the root.";
                  }

                  list reachability {
                    key "address interface";
                    description
                      "A next hop for reachability to root,
                       as a RIB view.";
                    leaf address {
                      type inet:ipv4-address;
                      description
                        "The next hop address to reach root.";
                    }
                    leaf interface {
                      type mpls-interface-ref;
                      description
                        "Interface connecting to next-hop.";
                    }
                    leaf peer {
                      type leafref {
                        path
                          "../../../../../../../../../peers/peer/"
                          + "lsr-id";
                      }
                      description
                        "LDP peer from which this next hop can be
                         reached.";
                    }
                  }
                } // list root
              } // roots
              container bindings {
                description
                  "mLDP FEC to label bindings.";
                container opaque-type-lspid {
                  description
                    "The type of opaque value element is
                     the generic LSP identifier";
                  reference
                    "RFC6388: Label Distribution Protocol
                     Extensions for Point-to-Multipoint and
                     Multipoint-to-Multipoint Label Switched
                     Paths.";
                  list fec-label {
                    key
                      "root-address lsp-id "
                      + "recur-root-address recur-rd";
                    description
```

```
            "List of FEC to label bindings.";
          leaf root-address {
            type inet:ipv4-address;
            description
              "Root address.";
          }
          leaf lsp-id {
            type uint32;
            description "ID to identify the LSP.";
          }
          leaf recur-root-address {
            type inet:ip-address;
            description
              "Recursive root address.";
            reference
              "RFC6512: Using Multipoint LDP When the
               Backbone Has No Route to the Root";
          }
          leaf recur-rd {
            type route-distinguisher;
            description
              "Route Distinguisher in the VPN-Recursive
               Opaque Value.";
            reference
              "RFC6512: Using Multipoint LDP When the
               Backbone Has No Route to the Root";
          }
          uses mldp-binding-label-state-attributes;
        } // fec-label
      } // opaque-type-lspid

      container opaque-type-src {
        description
          "The type of opaque value element is
           the transit source TLV";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "root-address source-address group-address "
            + "rd recur-root-address recur-rd";
          description
            "List of FEC to label bindings.";
          leaf root-address {
            type inet:ipv4-address;
```

```
            description
              "Root address.";
            }
            leaf source-address {
              type inet:ip-address;
              description
                "Source address.";
            }
            leaf group-address {
              type inet:ip-address-no-zone;
              description
                "Group address.";
            }
            leaf rd {
              type route-distinguisher;
              description
                "Route Distinguisher.";
              reference
                "RFC7246: Multipoint Label Distribution
                 Protocol In-Band Signaling in a Virtual
                 Routing and Forwarding (VRF) Table
                 Context.";
            }
            leaf recur-root-address {
              type inet:ip-address;
              description
                "Recursive root address.";
              reference
                "RFC6512: Using Multipoint LDP When the
                 Backbone Has No Route to the Root";
            }
            leaf recur-rd {
              type route-distinguisher;
              description
                "Route Distinguisher in the VPN-Recursive
                 Opaque Value.";
              reference
                "RFC6512: Using Multipoint LDP When the
                 Backbone Has No Route to the Root";
            }
            uses mldp-binding-label-state-attributes;
          } // fec-label
        } // opaque-type-src

        container opaque-type-bidir {
          description
            "The type of opaque value element is
             the generic LSP identifier";
```

```
                reference
                  "RFC6826: Multipoint LDP In-Band Signaling for
                   Point-to-Multipoint and
                   Multipoint-to-Multipoint Label Switched
                   Paths.";
                list fec-label {
                  key
                    "root-address rp group-address "
                    + "rd recur-root-address recur-rd";
                  description
                    "List of FEC to label bindings.";
                  leaf root-address {
                    type inet:ipv4-address;
                    description
                      "Root address.";
                  }
                  leaf rp {
                    type inet:ip-address;
                    description
                      "RP address.";
                  }
                  leaf group-address {
                    type inet:ip-address-no-zone;
                    description
                      "Group address.";
                  }
                  leaf rd {
                    type route-distinguisher;
                    description
                      "Route Distinguisher.";
                    reference
                      "RFC7246: Multipoint Label Distribution
                       Protocol In-Band Signaling in a Virtual
                       Routing and Forwarding (VRF) Table
                       Context.";
                  }
                  leaf recur-root-address {
                    type inet:ip-address;
                    description
                      "Recursive root address.";
                    reference
                      "RFC6512: Using Multipoint LDP When the
                       Backbone Has No Route to the Root";
                  }
                  leaf recur-rd {
                    type route-distinguisher;
                    description
                      "Route Distinguisher in the VPN-Recursive
```

```
                        Opaque Value.";
                      reference
                        "RFC6512: Using Multipoint LDP When the
                         Backbone Has No Route to the Root";
                    }
                    uses mldp-binding-label-state-attributes;
                  } // fec-label
                } // opaque-type-bidir
              } // bindings
            } // ipv4

            container ipv6 {
              when "../../afi = 'ipv6'" {
                description
                  "Only for IPv6.";
              }
              description
                "IPv6 state information.";
              container roots {
                description
                  "IPv6 multicast LSP roots.";
                list root {
                  key "root-address";
                  description
                    "List of roots for configured multicast LSPs.";

                  leaf root-address {
                    type inet:ipv6-address;
                    description
                      "Root address.";
                  }

                  leaf is-self {
                    type boolean;
                    description
                      "This is the root.";
                  }

                  list reachability {
                    key "address interface";
                    description
                      "A next hop for reachability to root,
                       as a RIB view.";
                    leaf address {
                      type inet:ipv6-address;
                      description
                        "The next hop address to reach root.";
                    }
```

```
            leaf interface {
              type mpls-interface-ref;
              description
                "Interface connecting to next-hop.";
            }
            leaf peer {
              type leafref {
                path
                  "../../../../../../../../../peers/peer/"
                  + "lsr-id";
              }
              description
                "LDP peer from which this next hop can be
                 reached.";
            }
          }
        } // list root
      } // roots
      container bindings {
        description
          "mLDP FEC to label bindings.";
        container opaque-type-lspid {
          description
            "The type of opaque value element is
             the generic LSP identifier";
          reference
            "RFC6388: Label Distribution Protocol
             Extensions for Point-to-Multipoint and
             Multipoint-to-Multipoint Label Switched
             Paths.";
          list fec-label {
            key
              "root-address lsp-id "
              + "recur-root-address recur-rd";
            description
              "List of FEC to label bindings.";
            leaf root-address {
              type inet:ipv6-address;
              description
                "Root address.";
            }
            leaf lsp-id {
              type uint32;
              description "ID to identify the LSP.";
            }
            leaf recur-root-address {
              type inet:ip-address;
              description
```

```
                    "Recursive root address.";
                  reference
                    "RFC6512: Using Multipoint LDP When the
                     Backbone Has No Route to the Root";
                }
                leaf recur-rd {
                  type route-distinguisher;
                  description
                    "Route Distinguisher in the VPN-Recursive
                     Opaque Value.";
                  reference
                    "RFC6512: Using Multipoint LDP When the
                     Backbone Has No Route to the Root";
                }
                uses mldp-binding-label-state-attributes;
              } // fec-label
            } // opaque-type-lspid

            container opaque-type-src {
              description
                "The type of opaque value element is
                 the transit Source TLV";
              reference
                "RFC6826: Multipoint LDP In-Band Signaling for
                 Point-to-Multipoint and
                 Multipoint-to-Multipoint Label Switched
                 Paths.";
              list fec-label {
                key
                  "root-address source-address group-address "
                  + "rd recur-root-address recur-rd";
                description
                  "List of FEC to label bindings.";
                leaf root-address {
                  type inet:ipv6-address;
                  description
                    "Root address.";
                }
                leaf source-address {
                  type inet:ip-address;
                  description
                    "Source address.";
                }
                leaf group-address {
                  type inet:ip-address-no-zone;
                  description
                    "Group address.";
                }
```

```
            leaf rd {
              type route-distinguisher;
              description
                "Route Distinguisher.";
              reference
                "RFC7246: Multipoint Label Distribution
                 Protocol In-Band Signaling in a Virtual
                 Routing and Forwarding (VRF) Table
                 Context.";
            }
            leaf recur-root-address {
              type inet:ip-address;
              description
                "Recursive root address.";
              reference
                "RFC6512: Using Multipoint LDP When the
                 Backbone Has No Route to the Root";
            }
            leaf recur-rd {
              type route-distinguisher;
              description
                "Route Distinguisher in the VPN-Recursive
                 Opaque Value.";
              reference
                "RFC6512: Using Multipoint LDP When the
                 Backbone Has No Route to the Root";
            }
            uses mldp-binding-label-state-attributes;
          } // fec-label
        } // opaque-type-src

        container opaque-type-bidir {
          description
            "The type of opaque value element is
             the generic LSP identifier";
          reference
            "RFC6826: Multipoint LDP In-Band Signaling for
             Point-to-Multipoint and
             Multipoint-to-Multipoint Label Switched
             Paths.";
          list fec-label {
            key
              "root-address rp group-address "
              + "rd recur-root-address recur-rd";
            description
              "List of FEC to label bindings.";
            leaf root-address {
              type inet:ipv6-address;
```

```
                        description
                          "Root address.";
                      }
                      leaf rp {
                        type inet:ip-address;
                        description
                          "RP address.";
                      }
                      leaf group-address {
                        type inet:ip-address-no-zone;
                        description
                          "Group address.";
                      }
                      leaf rd {
                        type route-distinguisher;
                        description
                          "Route Distinguisher.";
                        reference
                          "RFC7246: Multipoint Label Distribution
                           Protocol In-Band Signaling in a Virtual
                           Routing and Forwarding (VRF) Table
                           Context.";
                      }
                      leaf recur-root-address {
                        type inet:ip-address;
                        description
                          "Recursive root address.";
                        reference
                          "RFC6512: Using Multipoint LDP When the
                           Backbone Has No Route to the Root";
                      }
                      leaf recur-rd {
                        type route-distinguisher;
                        description
                          "Route Distinguisher in the VPN-Recursive
                           Opaque Value.";
                        reference
                          "RFC6512: Using Multipoint LDP When the
                           Backbone Has No Route to the Root";
                      }
                      uses mldp-binding-label-state-attributes;
                    } // fec-label
                  } // opaque-type-bidir
                } // bindings
              } // ipv6
            } // state

            container configured-leaf-lsps {
```

```
            description
              "Configured multicast LSPs.";

            container p2mp {
              description
                "Configured point-to-multipoint LSPs.";
              uses mldp-configured-lsp-roots;
            }
            container mp2mp {
              description
                "Configured multipoint-to-multipoint LSPs.";
              uses mldp-configured-lsp-roots;
            }
          } // configured-leaf-lsps
        } // list address-family
      } // mldp

      list address-family {
        key "afi";
        description
          "Per-vrf per-af params.";
        leaf afi {
          type ldp-address-family;
          description
            "Address family type value.";
        }

        container config {
          description
            "Configuration data.";
          leaf enable {
            type boolean;
            description
              "'true' to enable the address family.";
          }
          uses policy-container;

          container ipv4 {
            when "../../afi = 'ipv4'" {
              description
                "Only for IPv4.";
            }
            description
              "IPv4 address family.";
            leaf transport-address {
              type inet:ipv4-address;
              description
                "The transport address advertised in LDP Hello
```

```
                messages.";
            }
          } // ipv4
          container ipv6 {
            when "../../afi = 'ipv6'" {
              description
                "Only for IPv6.";
            }
            description
              "IPv6 address family.";
            leaf transport-address {
              type inet:ipv6-address;
              description
                "The transport address advertised in LDP Hello
                 messages.";
            }
          } // ipv6
        }
        container state {
          config false;
          description
            "Operational state data.";
          leaf enable {
            type boolean;
            description
              "'true' to enable the address family.";
          }

          uses policy-container;

          container ipv4 {
            when "../../afi = 'ipv4'" {
              description
                "Only for IPv4.";
            }
            description
              "IPv4 address family.";
            leaf transport-address {
              type inet:ipv4-address;
              description
                "The transport address advertised in LDP Hello
                 messages.";
            }

            container bindings {
              description
                "LDP address and label binding information.";
              list address {
```

```
                key "address";
                description
                  "List of address bindings.";
                leaf address {
                  type inet:ipv4-address;
                  description
                    "Binding address.";
                }
                uses binding-address-state-attributes;
              } // binding-address

              list fec-label {
                key "fec";
                description
                  "List of label bindings.";
                leaf fec {
                  type inet:ipv4-prefix;
                  description
                    "Prefix FEC.";
                }
                uses binding-label-state-attributes;
              } // fec-label
            } // binding
          } // ipv4
          container ipv6 {
            when "../../afi = 'ipv6'" {
              description
                "Only for IPv6.";
            }
            description
              "IPv6 address family.";
            leaf transport-address {
              type inet:ipv6-address;
              description
                "The transport address advertised in LDP Hello
                 messages.";
            }

            container binding {
              description
                "LDP address and label binding information.";
              list address {
                key "address";
                description
                  "List of address bindings.";
                leaf address {
                  type inet:ipv6-address;
                  description
```

```
                    "Binding address.";
                }
                uses binding-address-state-attributes;
              } // binding-address

              list fec-label {
                key "fec";
                description
                  "List of label bindings.";
                leaf fec {
                  type inet:ipv6-prefix;
                  description
                    "Prefix FEC.";
                }
                uses binding-label-state-attributes;
              } // fec-label
            } // binding
          } // ipv6
        } // state
      } // address-family

      container discovery {
        description
          "Neibgbor discovery configuration.";

        container interfaces {
          description
            "A list of interfaces for basic descovery.";
          container config {
            description
              "Configuration data.";
            uses basic-discovery-timers;
          }
          container state {
            config false;
            description

              "Operational state data.";
            uses basic-discovery-timers;
          }

          list interface {
            key "interface";
            description
              "List of LDP interfaces.";
            leaf interface {
              type mpls-interface-ref;
              description
```

```
            "Interface.";
          }
          container config {
            description
              "Configuration data.";
            uses basic-discovery-timers {
              if-feature per-interface-timer-config;
            }
            leaf igp-synchronization-delay {
              if-feature per-interface-timer-config;
              type uint16 {
                range 3..60;
              }
              units seconds;
              description
                "Sets the interval that the LDP waits before
                 notifying the Interior Gateway Protocol (IGP)
                 that label exchange is completed so that IGP
                 can start advertising the normal metric for
                 the link.";
            }
          }
          container state {
            config false;
            description
              "Operational state data.";
            uses basic-discovery-timers {
              if-feature per-interface-timer-config;
            }
            leaf igp-synchronization-delay {
              if-feature per-interface-timer-config;
              type uint16 {
                range 3..60;
              }
              units seconds;
              description
                "Sets the interval that the LDP waits before
                 notifying the Interior Gateway Protocol (IGP)
                 that label exchange is completed so that IGP
                 can start advertising the normal metric for
                 the link.";
            }
            leaf next-hello {
              type uint16;
              units seconds;
              description "Time to send the next hello message.";
            }
          } // state
```

```
                list address-family {
                  key "afi";
                  description
                    "Per-vrf per-af params.";
                  leaf afi {
                    type ldp-address-family;
                    description
                      "Address family type value.";
                  }
                  container config {
                    description
                      "Configuration data.";
                    leaf enable {
                      type boolean;
                      description
                        "Enable the address family on the interface.";
                    }

                    container ipv4 {
                      must "/if:interfaces/if:interface"
                        + "[name = current()/../../../interface]/"
                        + "ip:ipv4" {
                        description
                          "Only if IPv4 is enabled on the interface.";
                      }
                      description
                        "IPv4 address family.";
                      leaf transport-address {
                        type union {
                          type enumeration {
                            enum "use-interface-address" {
                              description
                              "Use interface address as the transport
                               address.";
                            }
                          }
                          type inet:ipv4-address;
                        }
                        description
                          "IP address to be advertised as the LDP
                           transport address.";
                      }
                    }

                    container ipv6 {
                      must "/if:interfaces/if:interface"
                        + "[name = current()/../../../interface]/"
                        + "ip:ipv6" {
```

```
                    description
                      "Only if IPv6 is enabled on the interface.";
                  }
                  description
                    "IPv6 address family.";
                  leaf transport-address {
                    type union {
                      type enumeration {
                        enum "use-interface-address" {
                          description
                            "Use interface address as the transport
                             address.";
                        }
                      }
                      type inet:ipv4-address;
                    }
                    description
                      "IP address to be advertised as the LDP
                       transport address.";
                  }
                } // ipv6
              }
              container state {
                config false;
                description
                  "Operational state data.";
                leaf enable {
                  type boolean;
                  description
                    "Enable the address family on the interface.";
                }

                container ipv4 {
                  must "/if:interfaces/if:interface"
                    + "[name = current()/../../../interface]/"
                    + "ip:ipv4" {
                    description
                      "Only if IPv4 is enabled on the interface.";
                  }
                  description
                    "IPv4 address family.";
                  leaf transport-address {
                    type union {
                      type enumeration {

                        enum "use-interface-address" {
                          description
                            "Use interface address as the transport
```

```
                    address.";
                }
              }
              type inet:ipv4-address;
            }
            description
              "IP address to be advertised as the LDP
               transport address.";
          }

          list hello-adjacencies {
            key "adjacent-address";
            description "List of hello adjacencies.";

            leaf adjacent-address {
              type inet:ipv4-address;
              description
                "Neighbor address of the hello adjacency.";
            }

            uses adjacency-state-attributes;

            leaf peer {
              type leafref {
                path "../../../../../../../../peers/peer/"
                  + "lsr-id";
              }
              description
                "LDP peer from this adjacency.";
            }
          } // hello-adjacencies
        }
        container ipv6 {
          must "/if:interfaces/if:interface"
            + "[name = current()/../../../interface]/"
            + "ip:ipv6" {
            description
              "Only if IPv6 is enabled on the interface.";
          }
          description
            "IPv6 address family.";
          leaf transport-address {
            type union {
              type enumeration {
                enum "use-interface-address" {
                  description
                  "Use interface address as the transport
                   address.";
```

```
                        }
                      }
                      type inet:ipv4-address;
                    }
                    description
                      "IP address to be advertised as the LDP
                       transport address.";
                  }

                  list hello-adjacencies {
                    key "adjacent-address";
                    description "List of hello adjacencies.";

                    leaf adjacent-address {
                      type inet:ipv6-address;
                      description
                        "Neighbor address of the hello adjacency.";
                    }

                    uses adjacency-state-attributes;

                    leaf peer {
                      type leafref {
                        path "../../../../../../../../../peers/peer/"
                          + "lsr-id";
                      }
                      description
                        "LDP peer from this adjacency.";
                    }
                  } // hello-adjacencies
                } // ipv6
              }
            } // address-family
          } // list interface
        } // interfaces

        container targeted
        {
          description
            "A list of targeted neighbors for extended discovery.";
          container config {

            description
              "Configuration data.";
            uses extended-discovery-timers;
            uses extended-discovery-policy-attributes;
          }
          container state {
```

```
            config false;
            description
              "Operational state data.";
            uses extended-discovery-timers;
            uses extended-discovery-policy-attributes;
          }

          list address-family {
            key "afi";
            description
              "Per-af params.";
            leaf afi {
              type ldp-address-family;
              description
                "Address family type value.";
            }

            container state {
              config false;
              description
                "Operational state data.";

              container ipv4 {
                when "../../afi = 'ipv4'" {
                  description
                    "For IPv4.";
                }
                description
                  "IPv4 address family.";
                list hello-adjacencies {
                  key "local-address adjacent-address";
                  description "List of hello adjacencies.";

                  leaf local-address {
                    type inet:ipv4-address;
                    description
                      "Local address of the hello adjacency.";
                  }
                  leaf adjacent-address {
                    type inet:ipv4-address;
                    description
                      "Neighbor address of the hello adjacency.";
                  }

                  uses adjacency-state-attributes;

                  leaf peer {
                    type leafref {
```

```
                    path "../../../../../../../../peers/peer/"
                      + "lsr-id";
                  }
                  description
                    "LDP peer from this adjacency.";
                }
              } // hello-adjacencies
            } // ipv4

            container ipv6 {
              when "../../afi = 'ipv6'" {
                description
                  "For IPv6.";
              }
              description
                "IPv6 address family.";
              list hello-adjacencies {
                key "local-address adjacent-address";
                description "List of hello adjacencies.";

                leaf local-address {
                  type inet:ipv6-address;
                  description
                    "Local address of the hello adjacency.";
                }
                leaf adjacent-address {
                  type inet:ipv6-address;
                  description
                    "Neighbor address of the hello adjacency.";
                }

                uses adjacency-state-attributes;

                leaf peer {
                  type leafref {
                    path "../../../../../../../../peers/peer/"
                      + "lsr-id";
                  }
                  description
                    "LDP peer from this adjacency.";
                }
              } // hello-adjacencies
            } // ipv6
          } // state

          container ipv4 {
            when "../afi = 'ipv4'" {
              description
```

```
                    "For IPv4.";
                }
                description
                  "IPv4 address family.";
                list target {
                  key "adjacent-address";
                  description
                    "Targeted discovery params.";

                  leaf adjacent-address {
                    type inet:ipv4-address;
                    description
                      "Configures a remote LDP neighbor and enables
                       extended LDP discovery of the specified
                       neighbor.";
                  }
                  container config {
                    description
                      "Configuration data.";
                    leaf enable {
                      type boolean;
                      description
                        "Enable the target.";
                    }
                    leaf local-address {
                      type inet:ipv4-address;
                      description
                        "The local address.";
                    }
                  }
                  container state {
                    config false;
                    description
                      "Operational state data.";
                    leaf enable {
                      type boolean;
                      description
                        "Enable the target.";
                    }
                    leaf local-address {
                      type inet:ipv4-address;
                      description
                        "The local address.";
                    }
                  } // state
                }
              } // ipv4
              container ipv6 {
```

```
                when "../afi = 'ipv6'" {
                  description
                    "For IPv6.";
                }
                description
                  "IPv6 address family.";
                list target {
                  key "adjacent-address";
                  description
                    "Targeted discovery params.";

                  leaf adjacent-address {
                    type inet:ipv6-address;
                    description
                      "Configures a remote LDP neighbor and enables
                       extended LDP discovery of the specified
                       neighbor.";
                  }
                  container config {
                    description
                      "Configuration data.";
                    leaf enable {
                      type boolean;
                      description
                        "Enable the target.";
                    }
                    leaf local-address {
                      type inet:ipv6-address;
                      description
                        "The local address.";
                    }
                  }
                  container state {
                    config false;
                    description
                      "Operational state data.";
                    leaf enable {
                      type boolean;
                      description
                        "Enable the target.";
                    }
                    leaf local-address {
                      type inet:ipv6-address;
                      description
                        "The local address.";
                    }
                  } // state
                }
```

```
            } // ipv6
          } // address-family
        } // targeted
      } // discovery

      container forwarding-nexthop {
        if-feature forwarding-nexthop-config;
        description
          "Configuration for forwarding nexthop.";

        container interfaces {
          description
            "A list of interfaces on which forwarding is
             disabled.";

          list interface {
            key "interface";
            description
              "List of LDP interfaces.";
            leaf interface {
              type mpls-interface-ref;
              description
                "Interface.";
            }
            list address-family {
              key "afi";
              description
                "Per-vrf per-af params.";
              leaf afi {
                type ldp-address-family;
                description
                  "Address family type value.";
              }
              container config {
                description
                  "Configuration data.";
                leaf ldp-disable {
                  type boolean;
                  description
                    "Disable LDP forwarding on the interface.";
                }
                leaf mldp-disable {
                  if-feature mldp;
                  type boolean;
                  description
                    "Disable mLDP forwarding on the interface.";
                }
              }
```

```
                container state {
                  config false;
                  description
                    "Operational state data.";
                  leaf ldp-disable {
                    type boolean;
                    description
                      "Disable LDP forwarding on the interface.";
                  }
                  leaf mldp-disable {
                    if-feature mldp;

                    type boolean;
                    description
                      "Disable mLDP forwarding on the interface.";
                  }
                }
              } // address-family
            } // list interface
          } // interfaces
        } // forwarding-nexthop
        uses policy-container {
          if-feature all-af-policy-config;
        }
      } // global

      container peers {
        description
          "Peers configuration attributes.";

        container config {
          description
            "Configuration data.";
          uses peer-authentication {
            if-feature global-session-authentication;
          }
          uses peer-attributes;

          container session-downstream-on-demand {
            if-feature session-downstream-on-demand-config;
            description
              "Session downstream-on-demand attributes.";
            leaf enable {
              type boolean;
              description
                "'true' if session downstream-on-demand is enabled.";
            }
            leaf peer-list {
```

```
            type peer-list-ref;
            description
              "The name of a peer ACL.";
          }
        }
      }
      container state {
        config false;
        description
          "Operational state data.";
        uses peer-authentication {
          if-feature global-session-authentication;
        }
        uses peer-attributes;

        container session-downstream-on-demand {
          if-feature session-downstream-on-demand-config;
          description
            "Session downstream-on-demand attributes.";
          leaf enable {
            type boolean;
            description
              "'true' if session downstream-on-demand is enabled.";
          }
          leaf peer-list {
            type peer-list-ref;
            description
              "The name of a peer ACL.";
          }
        }
      }

      list peer {
        key "lsr-id";
        description
          "List of peers.";

        leaf lsr-id {
          type yang:dotted-quad;
          description "LSR ID.";
        }

        container config {
          description
            "Configuration data.";
          leaf admin-down {
            type boolean;
            default false;
```

```
            description
              "'true' to disable the peer.";
          }

          container capability {
            description
              "Per peer capability";
            container mldp {
              if-feature mldp;
              description
                "mLDP capabilities.";
              uses mldp-capabilities;
            }
          }

          uses peer-af-policy-container {
            if-feature all-af-policy-config;
          }

          uses peer-authentication;

          uses graceful-restart-attributes-per-peer {
            if-feature per-peer-graceful-restart-config;
          }

          uses peer-attributes {
            if-feature per-peer-session-attributes-config;
          }

          container address-family {
            description
              "Per-vrf per-af params.";
            container ipv4 {
              description
                "IPv4 address family.";
              uses peer-af-policy-container;
            }
            container ipv6 {
              description
                "IPv6 address family.";
              uses peer-af-policy-container;
            } // ipv6
          } // address-family
        }
        container state {
          config false;
          description
            "Operational state data.";
```

```
            leaf admin-down {
              type boolean;
              default false;
              description
                "'true' to disable the peer.";
            }

            container capability {
              description
                "Per peer capability";
              container mldp {
                if-feature mldp;
                description
                  "mLDP capabilities.";
                uses mldp-capabilities;
              }
            }

            uses peer-af-policy-container {
              if-feature all-af-policy-config;
            }

            uses peer-authentication;

            uses graceful-restart-attributes-per-peer {
              if-feature per-peer-graceful-restart-config;
            }

            uses peer-attributes {
              if-feature per-peer-session-attributes-config;
            }

            container address-family {
              description
                "Per-vrf per-af params.";
              container ipv4 {
                description
                  "IPv4 address family.";
                uses peer-af-policy-container;

                list hello-adjacencies {
                  key "local-address adjacent-address";
                  description "List of hello adjacencies.";

                  leaf local-address {
                    type inet:ipv4-address;
                    description
                      "Local address of the hello adjacency.";
```

```
                  }
                  leaf adjacent-address {
                    type inet:ipv4-address;
                    description
                      "Neighbor address of the hello adjacency.";
                  }

                  uses adjacency-state-attributes;

                  leaf interface {
                    type mpls-interface-ref;
                    description "Interface for this adjacency.";
                  }
                } // hello-adjacencies
              } // ipv4
              container ipv6 {
                description
                  "IPv6 address family.";
                uses peer-af-policy-container;

                list hello-adjacencies {
                  key "local-address adjacent-address";
                  description "List of hello adjacencies.";

                  leaf local-address {
                    type inet:ipv6-address;
                    description
                      "Local address of the hello adjacency.";
                  }
                  leaf adjacent-address {
                    type inet:ipv6-address;
                    description
                      "Neighbor address of the hello adjacency.";
                  }

                  uses adjacency-state-attributes;

                  leaf interface {
                    type mpls-interface-ref;
                    description "Interface for this adjacency.";
                  }
                } // hello-adjacencies
              } // ipv6
            } // address-family

            uses peer-state-derived;
          } // state
        } // list peer
```

```
      } // peers
    } // container mpls-ldp
  }

  /*
   * RPCs
   */
  rpc mpls-ldp-clear-peer {
    description
      "Clears the session to the peer.";
    input {
      leaf lsr-id {
        type union {
          type yang:dotted-quad;
          type uint32;
        }
        description
          "LSR ID of peer to be cleared. If this is not provided
           then all peers are cleared";
      }
    }
  }

  rpc mpls-ldp-clear-hello-adjacency {
    description
      "Clears the hello adjacency";
    input {
      container hello-adjacency {
        description
          "Link adjacency or targettted adjacency. If this is not
           provided then all hello adjacencies are cleared";
        choice hello-adjacency-type {
          description "Adjacency type.";
          case targeted {
            container targeted {
              presence "Present to clear targeted adjacencies.";
              description
                "Clear targeted adjacencies.";
              leaf target-address {
                type inet:ip-address;
                description
                  "The target address. If this is not provided then
                   all targeted adjacencies are cleared";
              }
            } // targeted
          }
          case link {
            container link {
```

```
              presence "Present to clear link adjacencies.";
              description
                "Clear link adjacencies.";
              leaf next-hop-interface {
                type mpls-interface-ref;
                description

                  "Interface connecting to next-hop. If this is not
                   provided then all link adjacencies are cleared.";
              }
              leaf next-hop-address {
                type inet:ip-address;
                must "../next-hop-interface" {
                  description
                    "Applicable when interface is specified.";
                }
                description
                  "IP address of next-hop. If this is not provided
                   then adjacencies to all next-hops on the given
                   interface are cleared.";
              } // next-hop-address
            } // link
          }
        }
      }
    }
  }

  rpc mpls-ldp-clear-peer-statistics {
    description
      "Clears protocol statistics (e.g. sent and received
       counters).";
    input {
      leaf lsr-id {
        type union {
          type yang:dotted-quad;
          type uint32;
        }
        description
          "LSR ID of peer whose statistic are to be cleared.
           If this is not provided then all peers statistics are
           cleared";
      }
    }
  }

  /*
   * Notifications
```

```
    */
   notification mpls-ldp-peer-event {

     description
       "Notification event for a change of LDP peer operational
        status.";
     leaf event-type {
       type oper-status-event-type;
       description "Event type.";
     }
     uses ldp-peer-ref;
   }

   notification mpls-ldp-hello-adjacency-event {
     description
       "Notification event for a change of LDP adjacency operational
        status.";
     leaf event-type {
       type oper-status-event-type;
       description "Event type.";
     }
     uses ldp-adjacency-ref;
   }

   notification mpls-ldp-fec-event {
     description
       "Notification event for a change of FEC status.";
     leaf event-type {
       type oper-status-event-type;
       description "Event type.";
     }
     uses ldp-fec-event;
   }

   notification mpls-mldp-fec-event {
     description
       "Notification event for a change of FEC status.";
     leaf event-type {
       type oper-status-event-type;
       description "Event type.";
     }
     uses mldp-fec-event;
   }
 }

 <CODE ENDS>
```

Figure 22

7. Security Considerations

The configuration, state, action and notification data defined using YANG data models in this document are likely to be accessed via the protocols such as NETCONF [RFC6241] etc.

Hence, YANG implementations MUST comply with the security requirements specified in section 15 of [RFC6020]. Additionally, NETCONF implementations MUST comply with the security requirements specified in sections 2.2, 2.3 and 9 of [RFC6241] as well as section 3.7 of [RFC6536].

8. IANA Considerations

This document does not extend LDP or mLDP base protocol specifiction and hence there are no IANA considerations.

Note to the RFC Editor: Please remove IANA section before the publication.

9. Acknowledgments

The authors would like to acknowledge Eddie Chami, Nagendra Kumar, Mannan Venkatesan, Pavan Beeram for their contribution to this document. We also acknowledge Ladislav Lhotka for his useful comments as the YANG Doctor.

10. References

10.1. Normative References

[I-D.ietf-netmod-routing-cfg]
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-22 (work in progress), July 2016.

[I-D.rtgyangdt-rtgwg-ni-model]
Berger, L., Hopps, C., Lindem, A., and D. Bogdanovic, "Network Instance Model", draft-rtgyangdt-rtgwg-ni-model-00 (work in progress), May 2016.

[I-D.saad-mpls-base-yang]
Raza, K., Gandhi, R., Liu, X., Beeram, V., Saad, T., Bryskin, I., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for MPLS Base", draft-saad-mpls-base-yang-00 (work in progress), May 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3478] Leelanivas, M., Rekhter, Y., and R. Aggarwal, "Graceful Restart Mechanism for Label Distribution Protocol", RFC 3478, DOI 10.17487/RFC3478, February 2003, <http://www.rfc-editor.org/info/rfc3478>.

[RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <http://www.rfc-editor.org/info/rfc5036>.

[RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <http://www.rfc-editor.org/info/rfc5331>.

[RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, DOI 10.17487/RFC5561, July 2009, <http://www.rfc-editor.org/info/rfc5561>.

[RFC5918] Asati, R., Minei, I., and B. Thomas, "Label Distribution Protocol (LDP) 'Typed Wildcard' Forward Equivalence Class (FEC)", RFC 5918, DOI 10.17487/RFC5918, August 2010, <http://www.rfc-editor.org/info/rfc5918>.

[RFC5919] Asati, R., Mohapatra, P., Chen, E., and B. Thomas, "Signaling LDP Label Advertisement Completion", RFC 5919, DOI 10.17487/RFC5919, August 2010, <http://www.rfc-editor.org/info/rfc5919>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <http://www.rfc-editor.org/info/rfc6388>.

[RFC6389] Aggarwal, R. and JL. Le Roux, "MPLS Upstream Label Assignment for LDP", RFC 6389, DOI 10.17487/RFC6389, November 2011, <http://www.rfc-editor.org/info/rfc6389>.

[RFC6512] Wijnands, IJ., Rosen, E., Napierala, M., and N. Leymann, "Using Multipoint LDP When the Backbone Has No Route to the Root", RFC 6512, DOI 10.17487/RFC6512, February 2012, <http://www.rfc-editor.org/info/rfc6512>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

[RFC6826] Wijnands, IJ., Ed., Eckert, T., Leymann, N., and M. Napierala, "Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6826, DOI 10.17487/RFC6826, January 2013, <http://www.rfc-editor.org/info/rfc6826>.

[RFC7060] Napierala, M., Rosen, E., and IJ. Wijnands, "Using LDP Multipoint Extensions on Targeted LDP Sessions", RFC 7060, DOI 10.17487/RFC7060, November 2013, <http://www.rfc-editor.org/info/rfc7060>.

[RFC7140] Jin, L., Jounay, F., Wijnands, IJ., and N. Leymann, "LDP Extensions for Hub and Spoke Multipoint Label Switched Path", RFC 7140, DOI 10.17487/RFC7140, March 2014, <http://www.rfc-editor.org/info/rfc7140>.

[RFC7246] Wijnands, IJ., Ed., Hitchen, P., Leymann, N., Henderickx, W., Gulko, A., and J. Tantsura, "Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context", RFC 7246, DOI 10.17487/RFC7246, June 2014, <http://www.rfc-editor.org/info/rfc7246>.

[RFC7438] Wijnands, IJ., Ed., Rosen, E., Gulko, A., Joorde, U., and J. Tantsura, "Multipoint LDP (mLDP) In-Band Signaling with Wildcards", RFC 7438, DOI 10.17487/RFC7438, January 2015, <http://www.rfc-editor.org/info/rfc7438>.

[RFC7552] Asati, R., Pignataro, C., Raza, K., Manral, V., and R. Papneja, "Updates to LDP for IPv6", RFC 7552, DOI 10.17487/RFC7552, June 2015, <http://www.rfc-editor.org/info/rfc7552>.

[RFC7715] Wijnands, IJ., Ed., Raza, K., Atlas, A., Tantsura, J., and Q. Zhao, "Multipoint LDP (mLDP) Node Protection", RFC 7715, DOI 10.17487/RFC7715, January 2016, <http://www.rfc-editor.org/info/rfc7715>.

10.2. Informative References

[I-D.ietf-rtgwg-policy-model] Shaikh, A., Shakir, R., D'Souza, K., and C. Chase, "Routing Policy Configuration Model for Service Provider Networks", draft-ietf-rtgwg-policy-model-01 (work in progress), April 2016.

[I-D.iwijnand-mpls-mldp-multi-topology] Wijnands, I. and K. Raza, "mLDP Extensions for Multi Topology Routing", draft-iwijnand-mpls-mldp-multi-topology-03 (work in progress), June 2013.

[I-D.openconfig-netmod-opstate] Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-01 (work in progress), July 2015.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <http://www.rfc-editor.org/info/rfc4364>.

[RFC7307] Zhao, Q., Raza, K., Zhou, C., Fang, L., Li, L., and D. King, "LDP Extensions for Multi-Topology", RFC 7307, DOI 10.17487/RFC7307, July 2014, <http://www.rfc-editor.org/info/rfc7307>.

Appendix A. Additional Contributors

Stephane Litkowski
Orange.
Email: stephane.litkowski@orange.com

Reshad Rahman
Cisco Systems Inc.
Email: rrahman@cisco.com

Danial Johari
Cisco Systems Inc.
Email: dajohari@cisco.com

Authors' Addresses

Kamran Raza
Cisco Systems, Inc.
Email: skraza@cisco.com

Rajiv Asati
Cisco Systems, Inc.
Email: rajiva@cisco.com

Sowmya Krishnaswamy
Cisco Systems, Inc.
Email: sowkrish@cisco.com

Xufeng Liu
Ericsson
Email: xliu@kuatrotech.com

Jeff Tantsura
Ericsson
Email: jeff.tantsura@ericsson.com

Santosh Esale
Juniper Networks
Email: sesale@juniper.net

Xia Chen
Huawei Technologies
Email: jescia.chenxia@huawei.com

Loa Andersson
Huawei Technologies
Email: loa@pi.nu

Himanshu Shah
Ciena Corporation
Email: hshah@ciena.com

Matthew Bocci
Alcatel-Lucent
Email: matthew.bocci@alcatel-lucent.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 12, 2016

T. Saad
K. Raza
R. Gandhi
Cisco Systems Inc
X. Liu
Ericsson
V. Beeram
Juniper Networks
H. Shah
Ciena
I. Bryskin
X. Chen
Huawei Technologies
R. Jones
Brocade
B. Wen
Comcast

May 11, 2016

A YANG Data Model for MPLS Static LSPs
draft-saad-mpls-static-yang-03

Abstract

This document contains the specification for the MPLS Static Label Switched Paths (LSPs) YANG model. The model allows for the provisioning of static LSP(s) on LER(s) and LSR(s) devices along a LSP path without the dependency on any signaling protocol. The MPLS Static LSP model augments the MPLS base YANG model with specific data to configure and manage MPLS Static LSP(s).

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on November 12, 2016.

Table of Contents

1. Introduction

This document describes a YANG data model for configuring and managing the Static LSPs feature. The model allows the configuration of LER and LSR devices with the necessary MPLS cross-connects or bindings to realize an end-to-end LSP service.

A static LSP is established by manually specifying incoming and outgoing MPLS label(s) and necessary forwarding information on each of the traversed Label Edge Router (LER) and Label Switched Router (LSR) devices (ingress, transit, or egress nodes) of the forwarding path.

For example, on an ingress LER device, the model is used to associate a specific Forwarding Equivalence Class (FEC) of packets- e.g. matching a specific IP prefix in a Virtual Routing or Forwarding (VRF) instance- to an MPLS outgoing label imposition, next-hop(s) and respective outgoing interface(s) to forward the packet. On an LSR device, the model is used to create a binding that swaps the incoming label with an outgoing label and forwards the packet on one or

multiple egress path(s). On an egress LER, it is used to create a binding that decapsulates the incoming MPLS label and performs forwarding based on the inner MPLS label (if present) or IP forwarding in the packet.

The MPLS Static LSP YANG model is defined in module "ietf-mpls-static" and augments the MPLS Base YANG model defined in module "ietf-mpls" in [I-D.saad-mpls-base-yang]. The approach described in [I-D.openconfig-netmod-opstate] is adopted to represent data pertaining to configuration intended, applied state and derived state data elements. Each container in the model holds a "config" and "state" sub-container. The "config" sub-container is used to represent the intended configurable parameters, and the state sub-container is used to represent both the applied configurable parameters and any derived state, such as counters or statistical information.

1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

The following terms are defined in [RFC6020]:

- augment,
- configuration data,
- data model,
- data node,
- feature,
- mandatory node,
- module,
- schema tree,
- state data,
- RPC operation.

1.2. MPLS Static LSPs Model Tree Diagram

The MPLS Static LSP tree diagram is shown in Figure 1.

```
module: ietf-mpls-static
augment /rt:routing/mpls:mpls:
   +--rw static-lsps
      +--rw static-lsp* [name]
         +--rw name      string
         +--rw config
         |  +--rw in-segment
         |  |  +--rw (type)?
         |  |     +--:(ip-prefix)
         |  |     |  +--rw ip-prefix?        inet:ip-prefix
         |  |     +--:(mpls-label)
         |  |        +--rw incoming-label?   mpls:mpls-label
         |  +--rw operation?            enumeration
         |  +--rw (out-segment)?
         |     +--:(simple-path)
         |     |  +--rw next-hop?             inet:ip-address
         |     |  +--rw outgoing-label?       mpls:mpls-label
         |     |  +--rw outgoing-interface?   if:interface-ref
         |     +--:(path-list)
         |        +--rw paths* [path-index]
         |           +--rw path-index            uint16
         |           +--rw backup-path-index?    uint16
         |           +--rw next-hop?             inet:ip-address
         |           +--rw outgoing-labels*      mpls:mpls-label
         |           +--rw outgoing-interface?   if:interface-ref
         |           +--rw loadshare?            uint16
         |           +--rw role?                 enumeration
         +--ro state
            +--ro in-segment
            |  +--ro (type)?
            |     +--:(ip-prefix)
            |     |  +--ro ip-prefix?        inet:ip-prefix
            |     +--:(mpls-label)
            |        +--ro incoming-label?   mpls:mpls-label
            +--ro operation?            enumeration
            +--ro (out-segment)?
               +--:(simple-path)
               |  +--ro next-hop?             inet:ip-address
               |  +--ro outgoing-label?       mpls:mpls-label
               |  +--ro outgoing-interface?   if:interface-ref
               +--:(path-list)
                  +--ro paths* [path-index]
                     +--ro path-index            uint16
                     +--ro backup-path-index?    uint16
```

```
                        +--ro next-hop?             inet:ip-address
                        +--ro outgoing-labels*      mpls:mpls-label
                        +--ro outgoing-interface?   if:interface-ref
                        +--ro loadshare?            uint16
                        +--ro role?                 enumeration
```

Figure 1: MPLS Static LSP tree diagram

1.3. MPLS Static LSP YANG Module

```
 <CODE BEGINS>file "ietf-mpls-static@2016-05-11.yang"

 module ietf-mpls-static {

   namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-static";

   prefix "mpls-static";

   import ietf-mpls {
     prefix mpls;
   }

   import ietf-routing {
     prefix "rt";
   }

   import ietf-inet-types {
     prefix inet;
   }

   import ietf-interfaces {
     prefix "if";
   }

   organization "IETF MPLS Working Group";

   contact
     "WG Web:   <http://tools.ietf.org/wg/mpls/>

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

      WG Chair: Loa Andersson
                <mailto:loa@pi.nu>

      WG Chair: Ross Callon
                <mailto:rcallon@juniper.net>

      WG Chair: George Swallow
```

```
                <mailto:swallow.ietf@gmail.com>

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

      Editor:   Kamran Raza
                <mailto:skraza@cisco.com>

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

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

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

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

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

      Editor:   Xia Chen
                <mailto:jescia.chenxia@huawei.com>

      Editor:   Raqib Jones
                <mailto:raqib@Brocade.com>

      Editor:   Bin Wen
                <mailto:Bin_Wen@cable.comcast.com>";

   description
      "This YANG module augments the 'ietf-routing' module with basic
      configuration and operational state data for MPLS static";

   revision "2016-05-11" {
     description
       "Latest revision:
         - Addressed MPLS-RT review comments";
     reference "RFC 3031: A YANG Data Model for Static MPLS LSPs";
   }

   grouping path-basic_config {
     description "common definitions for statics";

     leaf next-hop {
       type inet:ip-address;
```

```
      description "next hop IP address for the LSP";
    }

    leaf outgoing-label {
      type mpls:mpls-label;
      description
        "label value to push at the current hop for the
         LSP";
    }

    leaf outgoing-interface {
      type if:interface-ref;
      description
        "The outgoing interface";
    }

  }

  grouping path-properties_config {
    description
      "MPLS path properties";
    leaf path-index {
      type uint16;
      description
        "Path identifier";
    }

    leaf backup-path-index {
      type uint32;
      description
        "Backup path identifier";
    }

    leaf next-hop {
      type inet:ip-address;
      description
        "The address of the next-hop";
    }

    leaf-list outgoing-labels {
      type mpls:mpls-label;
      ordered-by user;
      description
        "The outgoing MPLS labels to impose";
    }

    leaf outgoing-interface {
      type if:interface-ref;
```

```
      description
        "The outgoing interface";
    }

    leaf loadshare {
      type uint16;
      description
        "This value is used to compute a loadshare to perform un-equal
         load balancing when multiple outgoing path(s) are specified. A
         share is computed as a ratio of this number to the total under
         all configured path(s).";
    }

    leaf role {
      type enumeration {
        enum PRIMARY {
          description
            "Path as primary traffic carrying";
        }
        enum BACKUP {
          description
            "Path acts as backup";
        }
        enum PRIMARY_AND_BACKUP {
          description
            "Path acts as primary and backup simultaneously";
        }
      }
      description
        "The MPLS path role";
    }
  }

  grouping static-lsp_config {
    description "common definitions for static LSPs";

    container in-segment {
      description
        "MPLS incoming segment";
      choice type {
        description
          "Basic FEC choice";
        case ip-prefix {
          leaf ip-prefix {
            type inet:ip-prefix;
            description "An IP prefix";
          }
        }
```

```
          case mpls-label {
            leaf incoming-label {
              type mpls:mpls-label;
              description "label value on the incoming packet";
            }
          }
        }
      }

      leaf operation {
        type enumeration {
          enum impose-and-forward {
            description
              "Operation impose outgoing label(s) and forward to
               next-hop";
          }
          enum pop-and-forward {
            description
              "Operation pop incoming label and forward to next-hop";
          }
          enum pop-impose-and-forward {
            description
              "Operation pop incoming label, impose one or more
               outgoing label(s) and forward to next-hop";
          }
          enum swap-and-forward {
            description
              "Operation swap incoming label, with outgoing label and
               forward to next-hop";
          }
          enum pop-and-lookup {
            description
              "Operation pop incoming label and perform a lookup";
          }
        }
        description
          "The MPLS operation to be executed on the incoming packet";
      }

      choice out-segment {
        description "The MPLS out-segment type choice";
        case simple-path {
          uses path-basic_config;
        }
        case path-list {
          list paths {
            key path-index;
            description
```

```
            "The list of MPLS paths associated with the FEC";
          uses path-properties_config;
        }
      }
    }
  }

  grouping static-lsp {
    description "grouping for top level list of static LSPs";
    container config {
      description
        "Holds the intended configuration";
      uses static-lsp_config;
    }
    container state {
      config false;
      description
        "Holds the state and inuse configuration";
      uses static-lsp_config;
    }
  }

  augment "/rt:routing/mpls:mpls" {
    description "Augmentations for MPLS Static LSPs";
    container static-lsps {
      description
        "Statically configured LSPs, without dynamic signaling";
      list static-lsp {
        key name;
        description "list of defined static LSPs";

        leaf name {
          type string;
          description "name to identify the LSP";
        }
        uses static-lsp;
      }
    }
  }
}

<CODE ENDS>
```

Figure 2: MPLS Static LSP YANG module

2. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-mpls-static XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-mpls-static namespace: urn:ietf:params:xml:ns:yang:ietf-mpls-static prefix: ietf-mpls-static reference: RFC3031

3. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

4. References

4.1. Normative References

[I-D.saad-mpls-base-yang]
Raza, K., Gandhi, R., Liu, X., Beeram, V., Saad, T., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for MPLS Base and Static LSPs", draft-saad-mpls-base-yang-00 (work in progress), November 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

4.2. Informative References

[I-D.openconfig-netmod-opstate] Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-01 (work in progress), July 2015.

Authors' Addresses

Tarek Saad
Cisco Systems Inc

Email: tsaad@cisco.com

Kamran Raza
Cisco Systems Inc

Email: skraza@cisco.com

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Xufeng Liu
Ericsson

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Himanshu Shah
Ciena

Email: hshah@ciena.com

Igor Bryskin
Huawei Technologies

Email: Igor.Bryskin@huawei.com

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones
Brocade

Email: raqib@Brocade.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

CCAMP Working Group
Internet-Draft
Intended status: Standards Track

Xian Zhang
Huawei
R. Jing
China Telecom
W. Jian
China Unicom
Jeong-dong Ryoo
ETRI
Y. Xu
CAICT
Daniel King
Lancaster University

Expires: September 05, 2016 March 07, 2016

YANG Models for the Northbound Interface of a Transport Network Controller: Requirements, Functions, and a List of YANG Models

draft-zhang-ccamp-transport-ctrlnorth-yang-00.txt

Abstract

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic opaquely across the server-layer network resources. A transport network may be constructed from equipment utilizing any of a number of different transport technologies such as the evolving optical transport infrastructure (Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH) and Optical Transport Network (OTN)) or packet transport as epitomized by the MPLS Transport Profile (MPLS-TP).

All transport networks have high benchmarks for reliability and operational simplicity. This suggests a common, technology-independent management/control paradigm that can be extended to represent and configure specific technology attributes.

This document describes the requirements facing transport networks in order to provide open interfaces for resource programmability and control/management automation. A list of existing and additional YANG models is provided to fulfill the functional requirements.

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on September 05, 2016.

Table of Contents

1. Introduction

A transport network is a server-layer network designed to provide connectivity services, or more advanced services like Virtual Private Networks (VPN) for a client-layer network to carry the client traffic opaquely across the server-layer network resources. It acts as a pipe provider for upper-layer networks, such as IP network and mobile networks.

Transport networks, such as Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH), Optical Transport Network (OTN), Wavelength Division Multiplexing (WDM), and flexi-grid networks, are often built using equipment from a single vendor and are managed using private interfaces to dedicated Element Management Systems (EMS) / Network Management Systems (NMS). All transport networks have high benchmarks for reliability and operational simplicity. This suggests a common, technology-independent management/control paradigm that is extended to represent and configure specific technology attributes.

The need of network providers to manage multi-vendor and multi-domain transport networks (where each domain is an island of equipment from a single supplier) has been further stressed by the expansion in network size. At the same time, applications such as data center interconnection require larger and more dynamic connectivity matrices. Therefore, transport networks face new challenges going beyond automatic provisioning of tunnel setup enabled by GMPLS (Generalized Multi-Protocol Label Switching) protocols to achieve automatic service provisioning, as well as

address opportunities enabled by partitioning the network through the process of resource slicing. With lower operational expenditure (OPEX) and capital expenditure (CAPEX) as the usual objectives, open interfaces to transport networks are considered by network providers as a way to meet these requirements. The concept of Software Defined Networking (SDN) leverages these ideas.

The YANG language [RFC6020] is currently the data modeling language of choice within the IETF and has been adopted by a number of industry-wide open management and control initiatives. YANG may be used to model both configuration and operational states; it is vendor-neutral and supports extensible APIs for control and management of elements.

This document analyzes typical scenarios that need transport network control/management openness, and lists functions desired to enable deployment. Moreover, a list of YANG models and their relationships have been identified that can help facilitate the deployment and operation of transport network open interfaces. Note that some of the models discussed meet the requirements described, and are already being developed in the IETF. Thus, this document provides a reference of existing models, and provides information of the missing ones which need further work.

2. Conventions used in this document

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

3. Terminology and Notations

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in the YANG data tree presented later in this draft is defined in [ietf-netmod-rfc6087bis]. They are provided below for reference.

- Brackets "[" and "]" enclose list keys.

- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).

- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

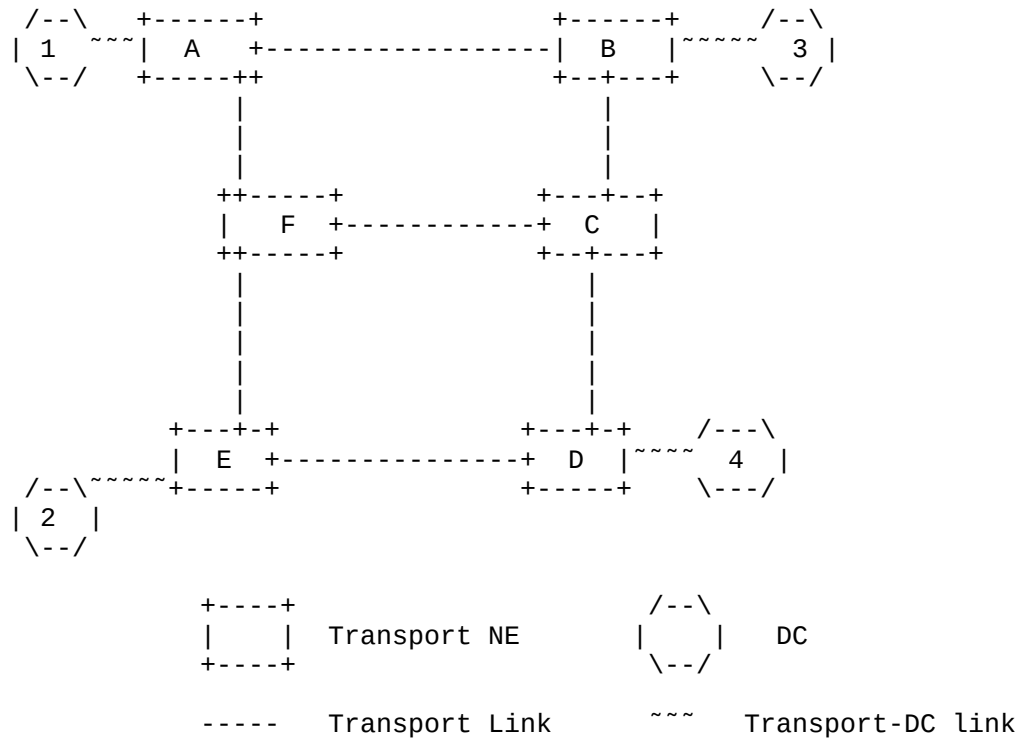
o Ellipsis ("...") stands for contents of subtrees that are not shown.

4. Functional Requirements

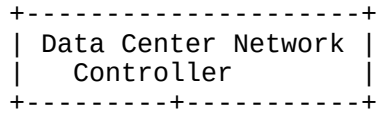
4.1. Scenarios

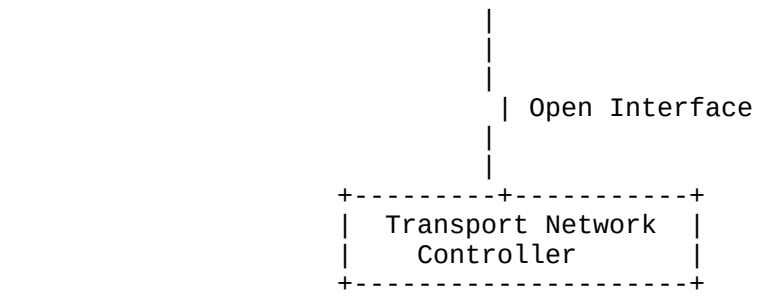
There are several scenarios where an open interface to access server-layer (transport) network resources would be useful. Here two typical scenarios are provided.

The first one is depicted as below (Figure 1):



(a) Data Centers interconnected via a transport network





(b) The controller architecture for data center interconnection

Figure 1: Scenario 1: Data centers interconnected via a transport network and the controller architecture

For the data center operator, assuming the objective is to trigger the transport network to provide connectivity on demand, the following capabilities, at a minimum, would be required on the open itnerface between the two contollers illustrated in Figure 1:

A: The ability to obtain information about a set of access points of the transport network facing the client side, including information such as access point identifiers, capabilities, etc.; for instance, transport-network-side end point identifiers related to the access link between DC1 and Transport NE A.

B: The capability to send a request for a service using the aforementioned access point information, as well as the ability to retrieve a list of service requests and their statuses. In this request, it should at least be possible to include source node, destination node, and requested bandwidth to request the transport network to set up tunnels/paths so as to provide the requested connectivitiy for the service request.

C: Note that in this case acquistion of the topology, be it physical or logical, of the transport network is not a compulsory requirement, but it may indeed be able to give data center providers more control over the transport resource usage. Furtheremore, the client controller can impose a virtual network of its own choice by requesting a slice of network resource with its choice of network parameters (such as network topology type, bandwidth etc.).

The second scenario, more complicated than the first, is depicted as below (Figure 2). In this example, we focus on the management and control via open interfaces for multi-domain networks with homogeneous technologies (such as OTN), but it can be extended

further to multi-domain networks with heterogeneous technologies with higher complexity.

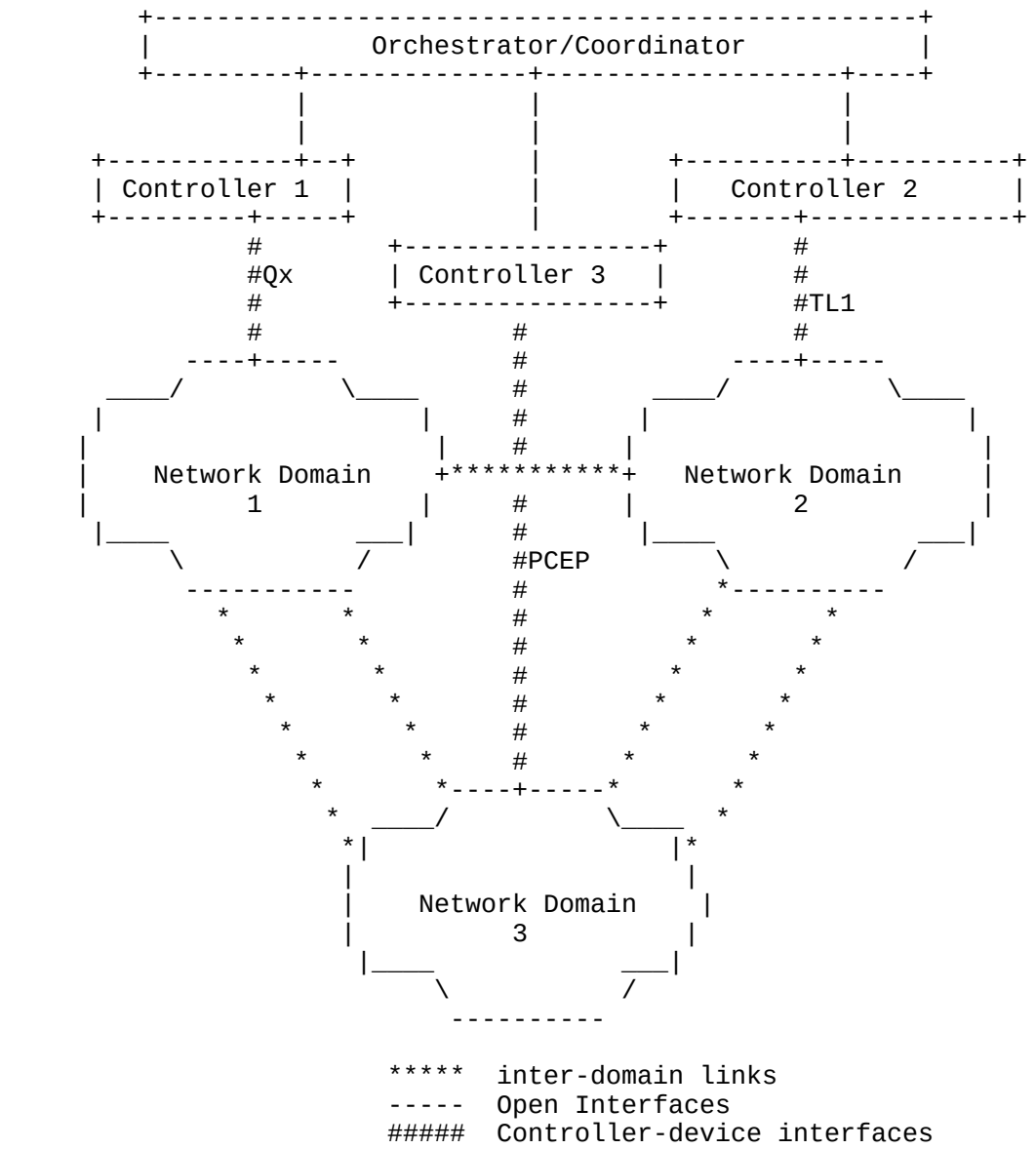


Figure 2: Scenario 2: Multi-domain network control and management

For the second scenario, the orchestrator/coordinator controls and manages three distinct network domains, each controlled/managed by their domain controller. In order to orchestrate across domains/layers, the orchestrator needs its interface between domain controllers to be equipped with the following functions:

A: Access to the topologies reported by each domain controller, including cross-domain links for the purpose of planning and requesting the paths of end-to-end tunnels. Multiple technologies within a domain (i.e., a multi-layer network), this might be reflected in the reported topology. Depending on the abstraction level of the reported topology, the orchestrator has different control granularities.

B: The ability to set up, delete and modify tunnels, be it within one domain or across multiple domains. Furthermore, it should have the abilty to view the tunnels created within each domain as well as thouse that cross domains as reported by each domain controller.

4.2. Function Requirement Summary

For the open interface of a transport controller towards a northbound client, five functions are derived from the scenarios explained in the last section. They are summarized in the table below and we also match these functions with YANG models that are being developed in existing drafts.
Analysis and descriptions of whether and how these functions are supported by the YANG models are provided in more detail in Section 5.

Functions	Description	Related Existing YANG Models
Obtaining Access Point Info	Getting the necessary access points info	[TE-Topo]
Obtaining Topology	Getting the topology info	[TE-Topo], [WDM-Topo] [ODU-Topo]
Tunnel Operations	Tunnel Setup, Deletion Modification and Info Retrieval	[TE-Tunnel]

Service Request	Requesting connectivity service and retrieval the list of service request	[this I.D.]	
Virtual Network Operations	Requesting a virtual network and related control operations, (e.g.,update, deletion)	[TE-Topo], [WDM topo] [ODU-Topo]	

5. Function Analysis

5.1. Toplogy Related Functions

As shown in Section 4, the functions of obtaining access point information, obtaining topology, and imposing virtual network operations can take advantages of the same set of topology YANG models. These functions are briefly explained further in the following sub-sections.

5.1.1 Obtaining Access Point Info

For cases such as scenario 1, a client may have no interest in directly controlling network resources, but might want an automated open control interface for initiating service requests. In this case, a transport controller may provide the access point information. This information can then be used in service request sent over the open interface.

The TE Topology YANG model provided in [TE-topo] can be used to provide a list of links. If the remote node and termination point information is unknown, it is omitted from the reported information. If the client-side node and termination point information is obtained via configuration or a distributed discovery mechanism, then it can also be added into the reported information. Technology-specific details might also be needed to further express the constraints/attributes associated with the access points. Note that all of this information is usually read only.

5.1.2 Obtaining Topology

Refer to [TE-Topo] for explanations and examples on how to obtain the topology. For technology specific topology information, other models such as those provided in [WDM-Topo] and [ODU-Topo] maybe used.

5.1.3 Virtual Network Operations

There are two ways to request the creation of a virtual network. One is to define the topology explicitly using the model provided in the topology YANG drafts listed in Section 5.1.2.. The other way is to provide an estimated traffic information (a traffic matrix) and ask for a network controller of the provider network to provide a virtual network that can fulfill the demand. This second approach does not have a supporting model and need further work.

5.2. Tunnel Operations

Thecurrent [TE-Tunnel] provides a technology agnostic Traffic-Engieering (TE) device tunnel. The model included in that draft is currently being developed to make it generic for both controller and device usage. It is expected that the next version of this draft will provide such a generic TE tunnel model that can cater to the base requirementss for tunnel operations but it may need to be augmented to support controller-specific operations.

Futhermore, technology-specific augmentations of the base generic TE tunnel models are needed. For example, for Optical Channel (OCh) tunnels in WDM networks, information such as the lambda resource usage is needed. Similarly, for ODU tunnels, information such as the usage of tributary slots is needed.

5.3. Service Requests

The service model is an important model that enables automated operations between a client controller and a provider controller. The transport connectivity service model is different from the model of a tunnel since the transport connectivity service model hides technical details from a client.

A transport connectivity service model is provided below:

```
module: ietf-transport-service
   +--rw transport_service
   |  +--rw service* [service-id]
   |     +--rw service-id           uint32
   |     +--rw service-name?        string
   |     +--rw source
   |     |  +--rw node-id?   node-id
   |     |  +--rw tp-id?     tp-id
   |     +--rw destination
   |     |  +--rw node-id?   node-id
```

```
|     |  +--rw tp-id?     tp-id
|     +--rw service-type?        service-types
|     +--rw supporting-tunnel* [name]
|     |  +--rw name    string
|     +--rw bandwidth            decimal64
|     +--rw SLA?                 SLAtypes
|     +--rw intended-policies
|        +--rw schedule
|           +--rw schedules
|              +--rw schedule* [schedule-id]
|                 +--rw schedule-id          uint32
|                 +--rw start?               yang:date-and-time
|                 +--rw schedule-duration?   string
|                 +--rw repeat-interval?     string
|
+--rw service-state
   +--ro service* [service-id]
      +--ro service-id           uint32
      +--ro service-name?        string
      +--ro source
      |  +--ro node-id?   node-id
      |  +--ro tp-id?     tp-id
      +--ro destination
      |  +--ro node-id?   node-id
      |  +--ro tp-id?     tp-id
      +--ro service-type?        service-types
      +--ro supporting-tunnel* [name]
      |  +--ro name    string
      +--ro bandwidth            decimal64
      +--ro SLA?                 SLAtypes
      +--ro applied-policies
      |  +--ro schedule
      |     +--ro schedules
      |        +--ro schedule* [schedule-id]
      |           +--ro schedule-id          uint32
      |           +--ro start?               yang:date-and-time
      |           +--ro schedule-duration?   string
      |           +--ro repeat-interval?     string
      +--ro status?              state-types
```

The corresponding YANG code is provided below:

<CODE BEGINS> file " ietf-transport-service@2016-3-7.yang"

```
module ietf-transport-service {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf_transport_service";
  prefix tser;
```

```
import ietf-inet-types {
  prefix inet;
}

import ietf-schedule {
  prefix "sch";
}

organization "TBD";
contact
  "WILL-BE-DEFINED-LATER";
description
  "this module describes a service module that is essential
  API for a client to ask for a provider network for a path
  without the need to care about underlying technologies.
  Capability to specify constraints/policies are provided as
  optional features.";

revision 2016-03-07 {
  description
    "Initial revision.";
    reference "to add the draft name";
}


typedef tp-id {  //client termination port
  type union {
    type uint32;
    type inet:ip-address; // IPv4 or IPv6 address
  }
  description
    "the client termination port of a transport device";
}

typedef node-id {  //client termination port
  type union {
    type uint32;
    type inet:ip-address; // IPv4 or IPv6 address
  }
  description
    "the node id of a transport device";
}

typedef service-types {
  type enumeration {
    enum "EPL" {
```

```
          value 0;
          description
          "EPL service";
        }
        enum "EVPL" {
          value 1;
          description
          "EVPL";
        }
        enum "EPLAN" {
          value 2;
          description
          "EPLAN";
        }
        enum "EVPLAN" {
          value 3;
          description
          "EVPLAN";
        }
      }
      description "the type of a service request";
    }

    typedef state-types{
      type enumeration {
        enum "NORMAL" {
          value 0;
          description
          "service is normal/up and running";
        }
        enum "DOWN" {
          value 1;
          description
          "service is down.";
        }
        enum "DEGRADED"{
          value 2;
          description
          "service is in degraded state.";
        }
      }
      description "the state of a service.";
    }

    typedef SLAtypes{
      type enumeration{
        enum "1+1+R"{
```

```
          value 0;
          description
          "A reroute will be provided after both the working and
          protection path fails.";
        }
        enum "1+1"{
          value 1;
          description
          "a protection path is provided.";
        }
        enum "Rerouting"{
          value 2;
          description
          "rerouting after the working path fails";
        }
        enum "unprotected"{
          value 3;
          description
          "no protection provided";
        }
      }
    }

    grouping service-basics {
      //later put all service under so that it can reused
      // in states.
      leaf service-id {
        type uint32;
        description "an unique identificaiton of a service.";
      }

      leaf service-name{
        type string;
        description "name for a service";
      }

      container source{
        leaf node-id {
          type node-id;
          description "node id";
        }
        leaf tp-id {
          type tp-id;
          description "TBD";
        }
        description "Service source information";
      }
```

```
      container destination{
        leaf node-id {
          type node-id;
          description "node id";
        }
        leaf tp-id {
          type tp-id;
          description "TBD";
        }
        description "Service destination information";
      }


      leaf service-type {
        type service-types;
        description "the type of a service request";
      }

      list supporting-tunnel{
        key "name";
        leaf name{
          type string;
          description "the name of a tunnel";
        }

        description "the list of tunnesl to support the list";
      }

      leaf bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        mandatory true;
        description "the bandwidth requested by a service.";
      }

      leaf SLA{
        type SLAtypes;
        description "the type of protection expected for this
        service";
      }
    }

    container transport_service {
      description
        "serves as a top-level container for a list of services";
```

```
      list service {
        key "service-id";
        description
          "an unique identifier of a service";

        uses service-basics;

        container intended-policies {
          container schedule {
            uses sch:schedules;
            description "to specify bandwidth scheduling
            information of this service.";
          }
          description "specify the policy associated with a
          service";
        }//end of policy
      }//end of service list
    }//service top container

    container service-state
    {
      list service {
        config false;
        key "service-id";
        description "operational state of a service";

        uses service-basics;

        container applied-policies{
          container schedule {
            uses sch:schedules;
            description "to specify bandwidth scheduling
            information of this service.";
          }
        }

        leaf status {
          type state-types;
          description "TBD";
        }
      }//end of a service state
    }//end of state
  }

  <CODE ENDS>
```

6. Security Considerations

Clearly modifying server-layer resources will have a significant impact on network infrastructure. More specifically they will provide the services and applications running across client-layers, which the server-layer is supporting. Therefore, security must be an important consideration when implementing the architecture, models and protocol mechanisms discussed in this document.

Communicating service and network information (including access point identifiers, capabilities, topologies, etc.) across external interfaces represents a security risk. Thus, mechanisms to encrypt or preserve the domain topology confidentiality should be used.

A key consideration are the external protocols (those shown as entering or leaving the orchestrator and controllers shown in Figure 2 (Scenario 2: Multi-domain network control and management)) which must be appropriately secured. This security should include authentication and authorization to control access to different functions that the orchestrator may perform to modify or create state in the server-layer, and the establishment and management of the orchestrator to controller relationship.

The orchestrator will contain significant data about the network domains, the services carried by each domain, and customer type information. Therefore, access to information held in the orchestrator must be secured. Since such access will be largely through external mechanisms, it may be pertinent to apply policy-based controls to restrict access and functions.

7. Manageability Considerations

The core objectives of this document are to assist in the deployment and operation of transport services across server-layer network infrastructure. The model-driven management/control principles, which are vendor-neutral and supported by extensible APIs, should be utilized.

The open models described in this document are based on YANG [RFC6020] and the RESTCONF [RESTCONF] messaging protocol, a REST-

like protocol running over HTTP for accessing data defined in YANG, may also be used.

8. IANA Considerations

TBD.

9. Acknowledgements

Thank Igor Bryskin for useful discussions on relevant YANG models.

10. References

10.1. Normative References

[RFC2119] S. Bradner, "Key words for use in RFCs to indicate requirements levels", RFC 2119, March 1997.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[ietf-netmod-rfc6087bis] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-01, work in progress, October 2014.

10.2. Informative References

[TE-Topo] Liu X., Bryskin I., et al, "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo-02, October 2015.

[WDM-Topo] Lee Y., et al, "A Yang Data Model for WSON Optical Networks", draft-lee-ccamp-wson-yang-02, work in progress, July 2015.

[ODU-Topo] Zhang X., Rao B., Liu X., "A YANG Data Model for Layer 1 Network Topology", draft-zhang-ccamp-l1-topo-yang-02, December 2015.

[TE-Tunnel] Saad T., Gandhi R., Liu X., et al, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-02, October, 2015.

[RESTCONF] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", Work in Progress, draft-ietf-netconf-restconf-09, December 2015.

11. Contributors' Address

Sergio Belotti
Nokia
Sergio.belotti@nokia.com

Young Lee
Futurewei Technologies
leeyoung@huawei.com

Aihua Guo
Huawei Technologies Canada
aihuaguo@huawei.com

Authors' Addresses

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

Ruiquan Jing
China Telecom
jingrq@ctbri.com.cn

Wei Jian
China Unicom
jianwei@chinaunicom.cn

Jeong-dong Ryoo
ETRI
ryoo@etri.re.kr

Yunbin Xu
China Academy of Information and Communication Technology (CAICT)
xuyunbin@ritt.cn

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

TEAS Working Group
Internet-Draft
Intended status: Experimental
Expires: May 25, 2017

Quintin Zhao
Robin Li
Boris Khasanov, Ed.
Huawei Technologies
King Ke
Tencent Holdings Ltd.
Luyuan Fang
Microsoft
Chao Zhou
Cisco Systems
Boris Zhang
Telus Communications
Artem Rachitskiy
Anton Gulida
Mobile TeleSystems JLLC

October 26, 2016

The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs
draft-zhao-teas-pcecc-use-cases-02

Abstract

In certain networks deployment scenarios, service providers would like to keep all the existing MPLS functionalities in both MPLS and GMPLS network while reducing existing complexity.In this document, we propose to use the PCE as a central controller so that LSP can be calculated/signaled/initiated/downloaded/managed through a centralized PCE server to each network devices along the LSP path while leveraging the existing PCE technologies as much as possible.

This draft describes the use cases for using the PCE as the central controller where LSPs are calculated/setup/initiated/downloaded/ maintained through extending the current PCE architectures and extending the PCEP.

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on May 25, 2017.

Table of Contents

1. Introduction

1.1. Background

In many network deployment scenarios, service providers would like to have the ability to dynamically adapt to a wide range of customer's requests for the sake of flexible network service delivery. SDN provides such flexibility and programmability for that case.

By migrating to the SDN enabled network from the existing network, service providers and network operators must have a solution which they can easily evolve from the existing network into the fully SDN enabled network while keeping scalability of the network services, guarantee robustness, availability, flexibility etc.

Taking into account the smooth transition from existing network to the new SDN enabled network with optimal cost, re-usage of the existing PCE components in network to be function of the central (SDN) controller is one choice, that not only achieves the goal of having centralized control but also leverages the existing PCE network components.

The Path Computation Element communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform route computations in response to Path Computation Clients (PCCs) requests. PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model draft [I-D. draft-ietf-pce-stateful-pce] describes a set of extensions to PCEP to enable active control of MPLS-TE and GMPLS tunnels.

[I-D.crabbe-pce-pce-initiated-lsp] describes the setup and teardown of PCE-initiated LSPs under the active stateful PCE model, without the need for local configuration on the PCC, thus allowing for a dynamic MPLS network that is centrally controlled and deployed.

[I-D.ali-pce-remote-initiated-gmpls-lsp] complements [I-D. draft-crabbe-pce-pce-initiated-lsp] by addressing the requirements for remote-initiated GMPLS LSPs.

Segment Routing (SR) technology leverages the source routing and tunneling paradigms. A source node can choose a path without relying on hop-by-hop signaling protocols such as LDP or RSVP-TE. Each path is specified as a set of "segments" advertised by link-state routing

protocols (IS-IS or OSPF). [I-D.filsfils-spring-segment-routing] provides an introduction to SR technology. The corresponding IS-IS and OSPF extensions are specified in [I-D.ietf-isis-segment-routing-extensions] and [I-D.psenak-ospf-segment-routing-extensions], respectively.

A Segment Routed path (SR path) can be derived from an IGP Shortest Path Tree (SPT). Segment Routed Traffic Engineering paths (SR-TE paths) may not follow IGP SPT. Such paths may be chosen by a suitable network planning tool and provisioned on the source node of the SR-TE path.

It is possible to use a stateful PCE for computing one or more SR-TE paths taking into account various constraints and objective functions. Once a path is chosen, the stateful PCE can instantiate an SR-TE path on a PCC using PCEP extensions specified in [I-D.crabbe-pce-pce-initiated-lsp] using the SR specific PCEP extensions described in [I-D.sivabalan-pce-segment-routing].

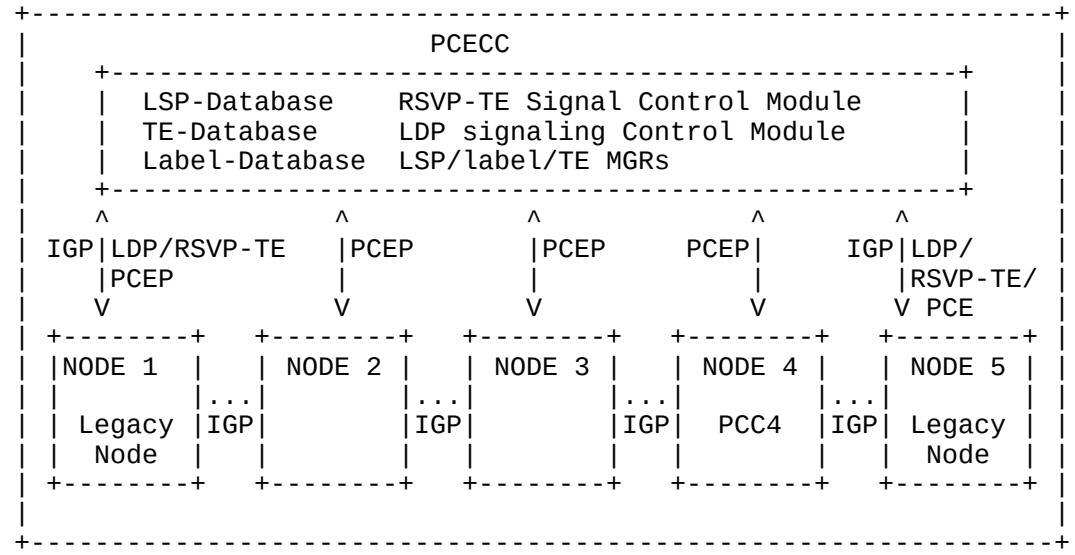
By using the solutions provided from above drafts, LSP in both MPLS and GMPLS network can be setup/delete/maintained/synchronized through a centrally controlled MPLS network.

The PCECC solution proposed in this document allows creation of dynamic MPLS network that is eventually controlled and deployed without the RSVP-TE protocol or extended IGP protocol with node/adjacency segment identifiers while providing all the key MPLS functionalities needed by the service providers.
These key MPLS features include MPLS P2P LSP, P2MP/MP2MP LSP, MPLS protection mechanism etc. In the case that one LSP path consists legacy network nodes and the new network nodes which are centrally controlled, the PCECC solution provides a smooth transition way for users.

1.2. Using the PCE as the Central Controller (PCECC) Approach

PCECC not only can remove the existing MPLS signaling totally from the control plane without losing any MPLS functionalities, but also will achieve this goal through utilizing the existing PCEP without introducing a new protocol into the network.

The following diagram illustrates the PCECC architecture.



Through the draft, we call the combination of the functionality for global label range signaling and the functionality of LSP setup/download/cleanup using the combination of global labels and local labels as PCECC functionality.

Current MPLS label has local meaning. That is, MPLS label allocated locally and signaled through the LDP/RSVP-TE/BGP etc. dynamic signaling protocol.

As the SDN(Service-Driven Network) technology develops, MPLS global label has been proposed again for new solutions. [I-D.li-mpls-global-label-usecases] proposes possible usecases of MPLS global label. MPLS global label can be used for identification of the location, the service and the network in different application scenarios. From these usecases we can see that no matter SDN or traditional application scenarios, the new solutions based on MPLS global label can facilitate service provisions.
The solution choices are described in [I-D.li-mpls-global-label-framework].

To ease the label allocation and signaling mechanism, also with the new applications such as concentrated LSP controller is introduced, PCE can be conveniently used as a central controller and MPLS global label range negotiator.

The later section of this draft describes the user cases for PCE server and PCE clients to have the global label range negotiation and local label range negotiation functionality.

To empower networking with centralized controllable modules, there are many choices for downloading the forwarding entries to the data plane, one way is the use of the OpenFlow protocol, which helps devices to populate their forwarding tables according to a set of instructions to the data plane. There are other candidate protocols to convey specific configuration information towards devices also. Since the PCEP protocol is already deployed in some of the service providers networks, leverage the PCEP to populated the MPLS forwarding table is a possible good choice.

For the centralized network, the performance achieved through distributed system can not be easy matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that the adjacency IDs for all the network nodes and links are propagated through the centralized controller instead of using the IGP extension.

The node and link adjacency IDs can be negotiated through the PCECC with each PCECC clients and these IDs can be just taken from the global label range which has been negotiated already.

With the capability of supporting SR within the PCECC architecture, all the p2p forwarding path protection use cases described in the draft [I-D.ietf-spring-resiliency-use-cases] will be supported too within the PCECC network. These protection alternatives include end-to-end path protection, local protection without operator management and local protection with operator management.

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP LSP using the local label range for each network nodes.

With the capability of setting up/maintaining the P2MP/MP2MP LSP within the PCECC network, it is easy to provide the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.

2. Terminology

The following terminology is used in this document.

IGP: Interior Gateway Protocol. Either of the two routing protocols, Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS).

PCC: Path Computation Client: any client application requesting a path computation to be performed by a Path Computation Element.

PCE: Path Computation Element. An entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

TE: Traffic Engineering.

3. PCEP Requirements

Following key requirements associated PCECC should be considered when designing the PCECC based solution:

1. Path Computation Element (PCE) clients supporting this draft MUST have the capability to advertise its PCECC capability to the PCECC.

2. Path Computation Element (PCE) supporting this draft MUST have the capability to negotiate a global label range for a group of clients.

3. Path Computation Client (PCC) MUST be able ask for global label range assigned in path request message .

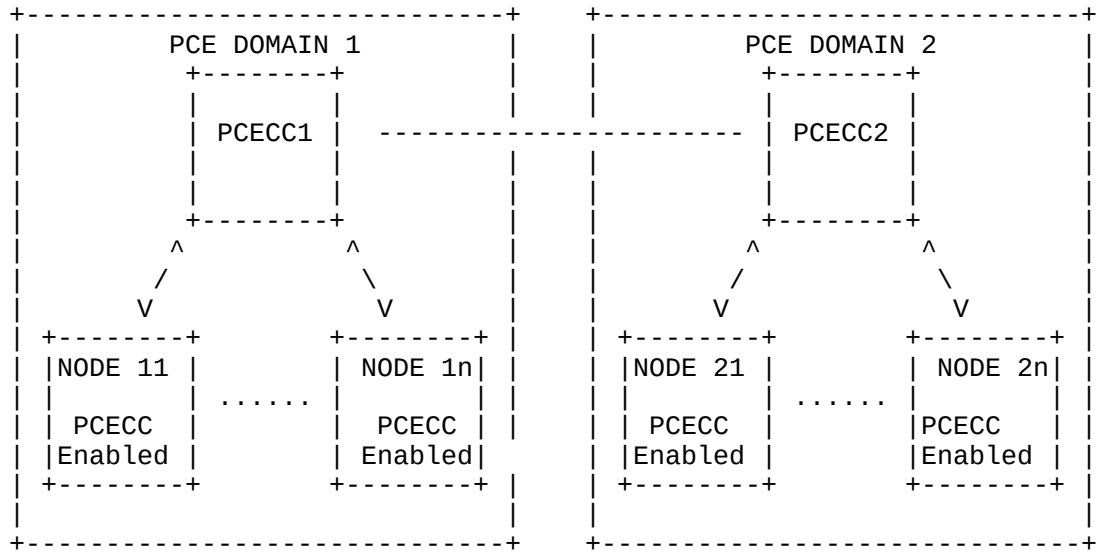
4. PCE are not required to support label reserve service. Therefore, it MUST be possible for a PCE to reject a Path Computation Request message with a reason code that indicates no support for label reserve service.

5. PCEP SHOULD provide a means to return global label range and LSP label assignments of the computed path in the reply message.

6. PCEP SHOULD provide a means to download the MPLS forwarding entry to the PCECC’s clients.

4. Use Cases of PCECC for Label Resource Reservations

Example 1 to 2 are based on network configurations illustrated using the following figure:



Example 1: Shared Global Label Range Reservation

- PCECC Clients nodes report MPLS label capability to the central controller PCECC.
- The central controller PCECC collects MPLS label capability of all nodes. Then PCECC can calculate the shared MPLS global label range for all the PCECC client nodes.
- In the case that the shared global label range need to be negotiated across multiple domains, the central controllers of these domains need to be communicate to negotiate a common global label range.
- The central controller PCECC notifies the shared global label range to all PCECC client nodes.

Example 2: Global Label Allocation

- PCECC Client node1 send global label allocation request to the central controller PCECC1.

- The central controller PCECC1 allocates the global label for FEC1 from the shared global label range and sends the reply to the client node1.

- The central controller PCECC1 notifies the allocated label for FEC1 to all PCECC client nodes within domain 1.

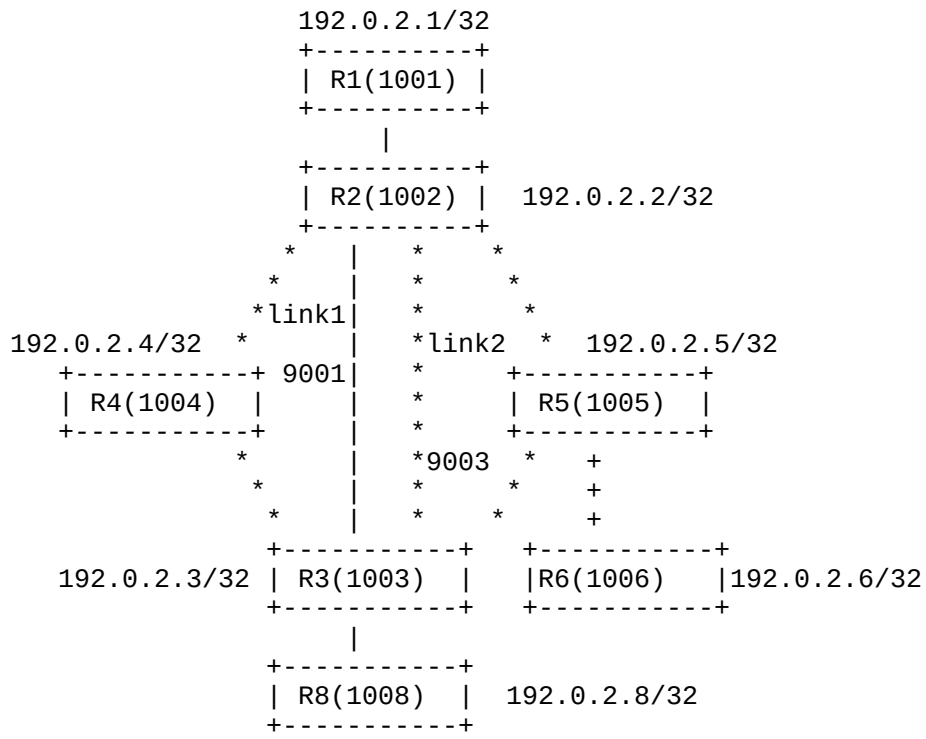
5. Using PCECC for SR without the IGP Extension

For the centralized network, the performance achieved through distributed system can not be easy matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that node segment ids and adjacency segment IDs for all the network are allocated dynamically and propagated through the centralized controller instead of using the IGP extension.

When the PCECC is used for the distribution of the node segment ID and adjacency segment ID, the node segment ID is allocated from the global label pool. For the allocation of adjacency segment ID, there are two choices, the first choice is that it is allocated from the local label pool, the second choice is that it is allocated from the global label pool. The advantage for the second choice is that the depth of the label stack for the forwarding path encoding will be reduced since adjacency segment ID can signal the forwarding path without adding the node segment ID in front of it. In this version of the draft, we use the fist choice for now. We may update the draft to reflect the use of the second choice.

Same as the SR solutions, when PCECC is used as the central controller, the support of FRR on any topology can be pre-computated and setup without any additional signaling (other than the regular IGP/BGP protocols) including the support of shared risk constraints, support of node and link protection and support of microloop avoidance.

The following example illustrate the use case where the node segment ID and adjacency segment ID are allocated from the global label allocated for SR path.



5.1. Use Cases of PCECC for SR Best Effort(BE) Path

In this mode of the solution, the PCECC just need to allocate the node segment ID and adjacency ID without calculating the explicit path for the SR path. The ingress of the forwarding path just need to encapsulate the destination node segment ID on top of the packet. All the intermediate nodes will forward the packet based on the final destination node segment id. It is similar to the LDP LSP forwarding except that label swapping is using the same global label both for the in segment and out segment in each hop.

The p2p SR BE path examples are explained as bellow:

Note that the node segment id for each node from the shared global labels ranges negotiated already.

Example 1:

R1 may send a packet to R8 simply by pushing an SR header with segment list {1008}. The path can be: R1-R2-R3-R8 or R1-R2-R5-R8 depending on the route calculation on node R2.

Example 2: local link/node protection:

For the packet which has destination of R3 and after that, R2 may preinstalled the backup forwarding entry to protect the R4 node, the pre-installed the backup path can go through either node5 or link1 or link2 between R2 and R3. The backup path calculation is locally decided by R2 and any existing IP FRR algorithms can be used here.

5.2. Use Cases of PCECC for SR Traffic Engineering (TE) Path

In the case of traffic engineering path is needed, the PCECC need to allocate the node segment ID and adjacency ID, and at the same time PCECC calculates the explicit path for the SR path and pass this explicit path represented with a sequence of node segment id and adjacency id. The ingress of the forwarding path need to encapsulate the stack of node segment id and adjacency id on top of the packet. For the case where strict traffic engineering path is needed, all the intermediate nodes and links will be specified through the stack of labels so that the packet is forwarded exactly as it is wanted.

Even though it is similar to TE LSP forwarding where forwarding path is engineered, but the Qos is only guaranteed through the enforce of the bandwidth admission control. As for the RSVP-TE LSP case, Qos is guaranteed through the link bandwidth reservation in each hop of the forwarding path.

The p2p SR traffic engineering path examples are explained as bellow:

Note that the node segment id for each node is allocated from the shared global labels ranges negotiated already and adjacency segment ids for each link are allocated from the local label pool for each node.

Example 1:

R1 may send a packet P1 to R8 simply by pushing an SR header with segment list {1008}. The path should be: R1-R2-R3-R8.

Example 2:

R1 may send a packet P2 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.

Example 3:

R1 may send a packet P3 to R8 while avoiding the links between R2 and R3 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8

The p2p local protection examples for SR TE path are explained as below:

Example 4: local link protection:

- R1 may send a packet P4 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.

- When node R2 receives the packet from R1 which has the header of R2- (1)link-R3-R8, and also find out there is a link failure of link1, then it will send out the packet with header of R3-R8 through link2.

Example 5: local node protection:

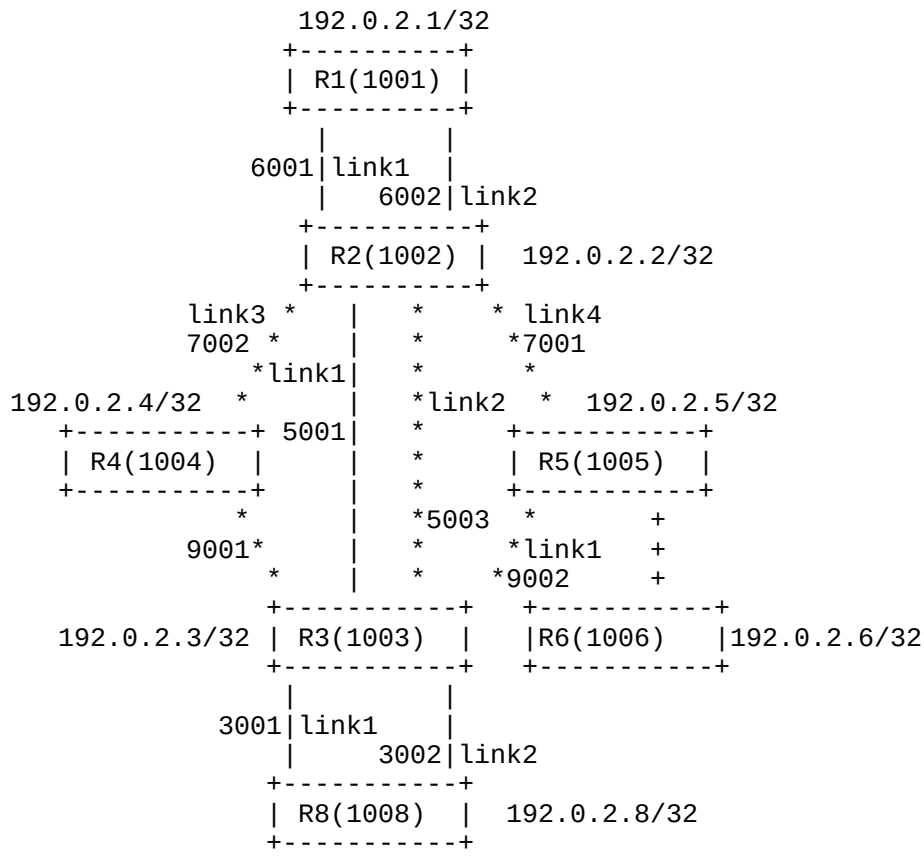
- R1 may send a packet P5 to R8 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8.

- When node R2 receives the packet from R1 which has the header of {1004, 1008}, and also find out there is a node failure for node4, then it will send out the packet with header of {1005, 1008} to node5 instead of node4.

6. Use Cases of PCECC for TE LSP

In the previous sections, we have discussed the cases where the SR path is setup through the PCECC. Although those cases give the simplicity and scalability, but there are existing functionalities for the traffic engineering path such as the bandwidth guarantee through the full forwarding path and the multicast forwarding path which SR based solution cannot solve. Also there are cases where the depth of the label stack may have been an issue for existing deployment and certain vendors.

So to address these issues, PCECC architecture should also support the TE LSP and multicast LSP functionalities. To achieve this, the existing PCEP can be used to communicate between the PCE server and PCE’s client PCC for exchanging the path request and reply information regarding to the TE LSP info. In this case, the TE LSP info is not only the path info itself, but it includes the full forwarding info. Instead of letting the ingress of LSP to initiate the LSP setup through the RSVP-TE signaling protocol, with minor extensions, we can use the PCEP to download the complete TE LSP forwarding entries for each node in the network.



TE LSP Setup Example

- Node1 sends a path request message for the setup of TE LSP from R1 to R8.

- PCECC program each node along the path from R1 to R8 with the primary path: {R1, link1, 6001}, {R2, link3, 7002], {R4, link0, 9001}, {R3, link1, 3001}, {R8}.

- For the end to end protection, PCECC program each node along the path from R1 to R8 with the secondary path: {R1, link2, 6002}, {R2, link4, 7001], {R5, link1, 9002}, {R3, link2, 3002}, {R8}.

- It is also possible to have a secondary backup path for the local node protection setup by PCECC. For example, the primary path is still same as what we have setup so far, then to protect the node R4 locally, PCECC can program the secondary path like this: {R1,

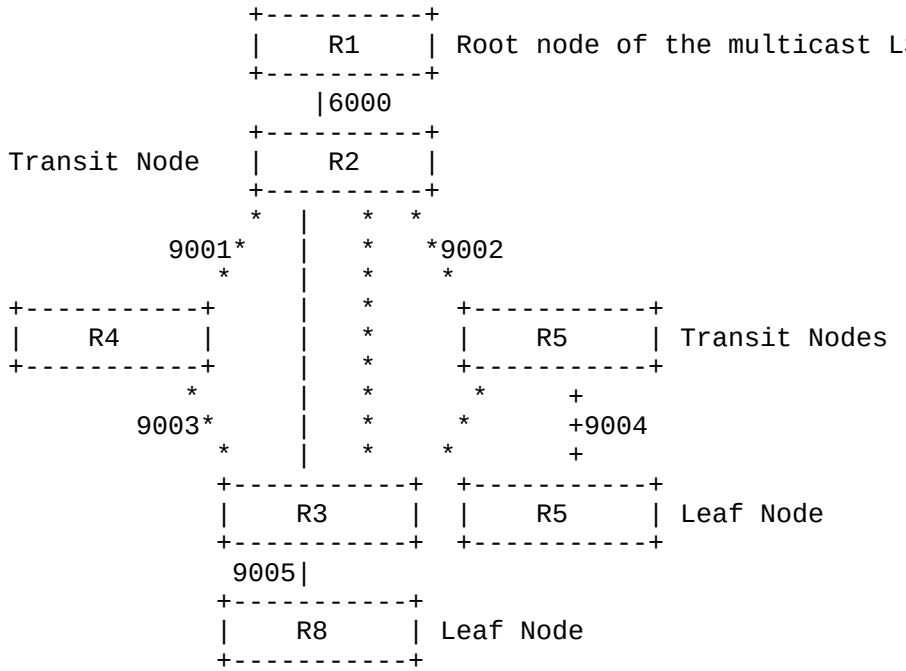
link1, 6001}, {R2, link1, 5001}, {R3, link1, 3001}, {R8}. By doing this, the node R4 is locally protected.

7. Use Cases of PCECC for Multicast LSPs

The current multicast LSPs are setup either using the RSVP-TE P2MP or mLDP protocols. The setup of these LSPs not only need a lot of manual configurations, but also it is also complex when the protection is considered. By using the PCECC solution, the multicast LSP can be computed and setup through centralized controller which has the full picture of the topology and bandwidth usage for each link. It not only reduces the complex configurations comparing the distributed RSVP-TE P2MP or mLDP signal lings, but also it can compute the disjoint primary path and secondary path efficiently.

7.1. Using PCECC for P2MP/MP2MP LSPs' Setup

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP lsp using the local label range for each network nodes.



The P2MP examples are explained here:

Step1: R1 may send a packet P1 to R2 simply by pushing an label of 6000 to the packet.

Step2: After R2 receives the packet with label 6000, it will forwarding to R4 by pushing header of 9001 and R5 by pusing header of 9002.

Step3: After R4 receives the packet with label 9001, it will forwarding to R3 by pushing header of 9003. After R5 receives the packet with label 9002, it will forwarding to R5 by pushing header of 9004.

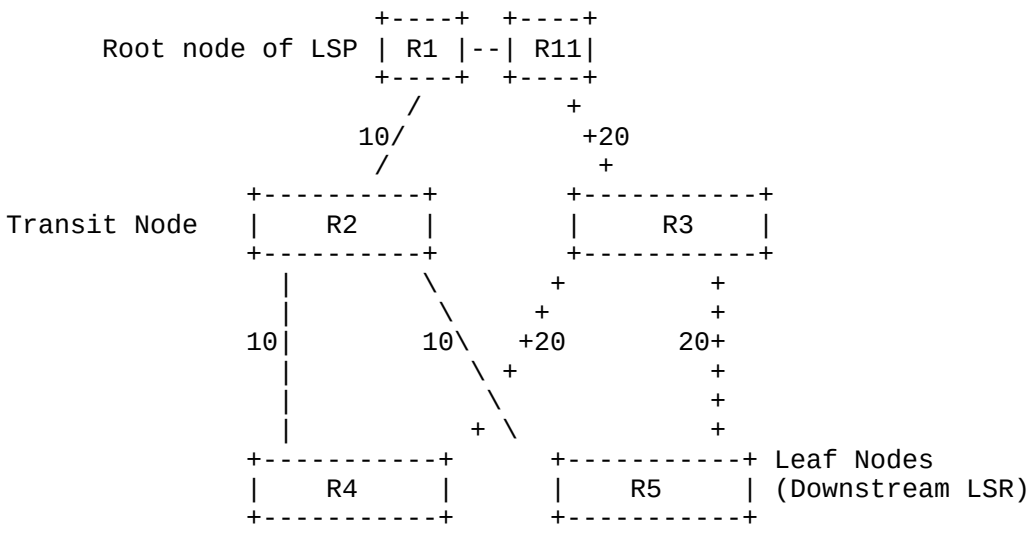
Step3: After R3 receives the packet with label 9003, it will forwarding to R8 by pushing header of 9005

7.2. Use Cases of PCECC for the Resiliency of P2MP/MP2MP LSPs

7.2.1. PCECC for the End-to-End Protection of the P2MP/MP2MP LSPs

In this section we describe the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.

An end-to-end protection (for nodes and links) principle can be applied for computing backup P2MP or MP2MP LSPs. During computation of the primarily multicast trees, PCECC server may also be taken into consideration to compute a secondary tree. A PCE may compute the primary and backup P2MP or MP2Mp LSP together or sequentially.

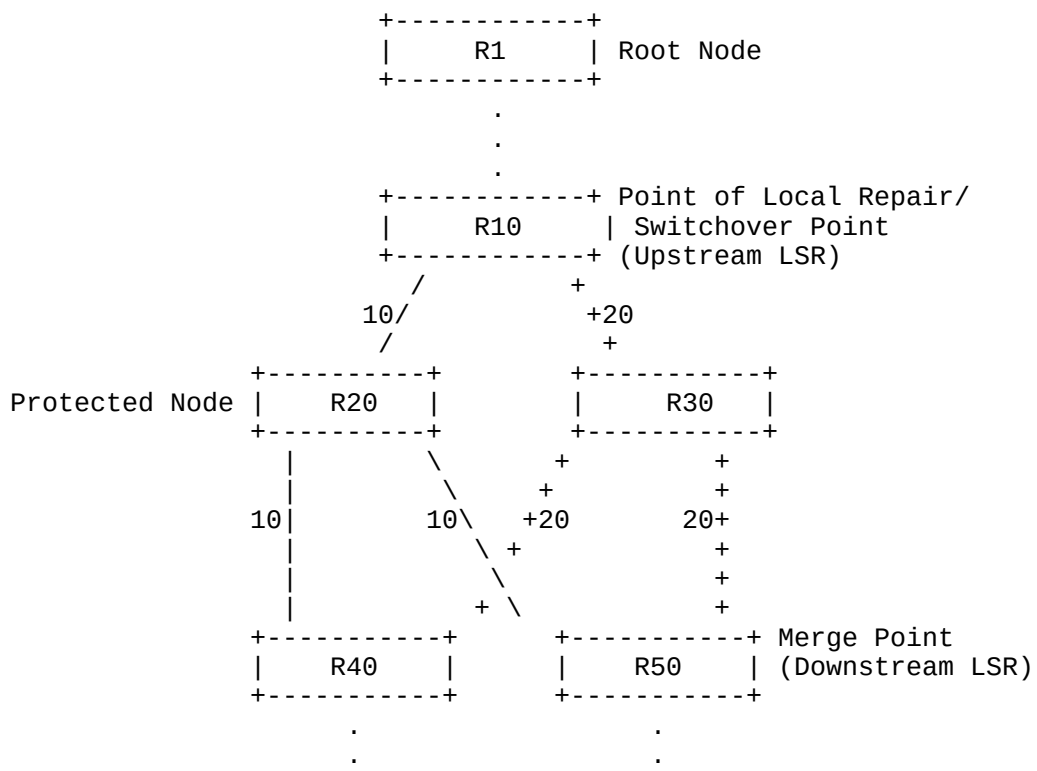


In the example above, when the PCECC setup the primary multicast tree from the root node R1 to the leafs, which is R1->R2->{R4, R5}, at same time, it can setup the backup tree, which is R11->R3->{R4, R5}. Both the these two primary forwarding tree and secondary forwarding tree will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R1->R2->{R4, R5} path normally, and when there is a node in the primary tree, then the root node R1 will switch the flow to the backup tree, which is R11->R3->{R4, R5}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

7.2.2. PCECC for the Local Protection of the P2MP/MP2MP LSPs

In this section we describe the local protection service in the PCECC network for the P2MP/MP2MP LSP.

While the PCECC sets up the primary multicast tree, it can also build the back LSP among PLR, the protected node, and MPs (the downstream nodes of the protected node). In the cases where the amount of downstream nodes are huge, this mechanism can avoid unnecessary packet duplication on PLR, so that protect the network from traffic congestion risk.



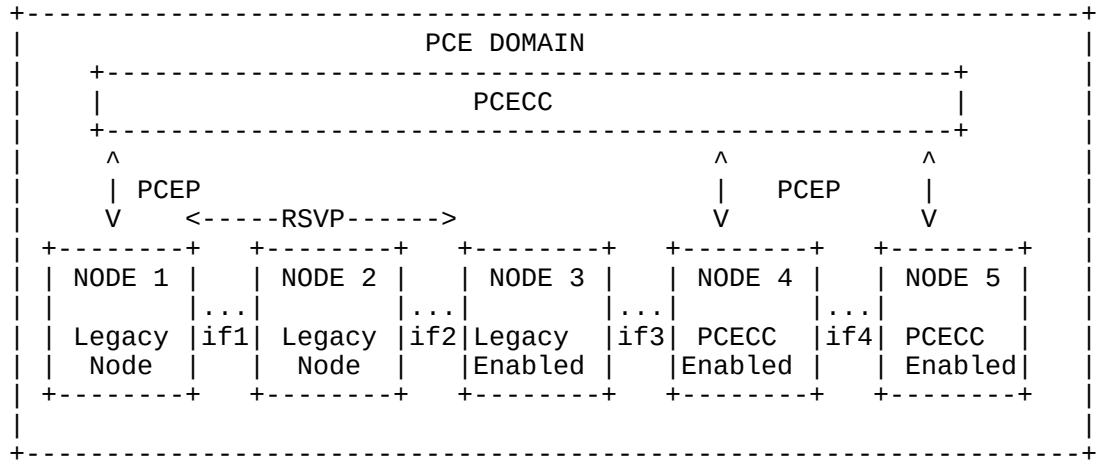
In the example above, when the PCECC setup the primary multicast path around the PLR node R10 to protect node R20, which is R10->R20->{R40, R50}, at same time, it can setup the backup path R10->R30->{R40, R50}. Both the these two primary forwarding path and secondary forwarding path will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R10->R20->{R40, R50} path normally, and when there is a node failure for node R20, then the PLR node R10 will switch the flow to the backup path, which is R10->R30->{R40, R50}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

8. Use Cases of PCECC for LSP in the Network Migration

One of the main advantages for PCECC solution is that it has backward compatibility naturally since the PCE server itself can function as a proxy node of MPLS network for all the new nodes which don’t support the existing MPLS signaling protocol anymore.

As it is illustrated in the following example, the current network will migrate to a total PCECC controlled network gradually by replacing the legacy nodes. During the migration, the legacy nodes still need to signal using the existing MPLS protocol such as LDP and RSVP-TE, and the new nodes setup their portion of the forwarding path through PCECC directly. With the PCECC function as the proxy of these new nodes, MPLS signaling can populate through network as normal.

Example described in this section is based on network configurations illustrated using the following figure:



Example: PCECC Initiated LSP Setup In the Network Migration

In this example, there are five nodes for the TE LSP from head end (Node1) to the tail end (Node5). Where the Node4 and Node5 are centrally controlled and other nodes are legacy nodes.

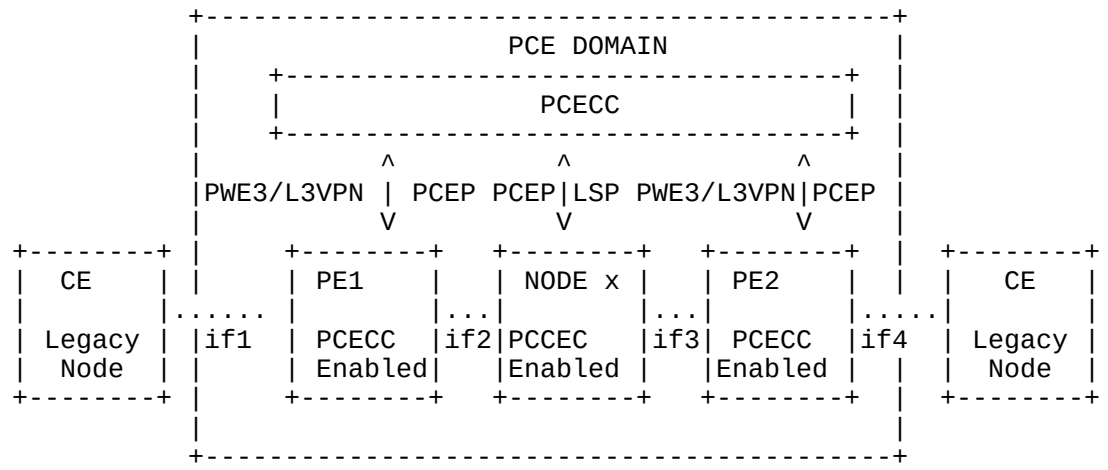
- Node1 sends a path request message towards PCECC for the setup of LSP destinating to Node5.
- PCECC sends to node1 a reply message for LSP setup with the path: (Node1, if1),(Node2, if2), (Node3, if3), (Node4, if4), Node5.
- Node1, Node2, Node3 will setup the LSP to Node5 using the local labels as usual.
- Then the PCECC will program the outsegment of Node3, the insegment/ ousegment of Node4, and the insegment for Node5.

9. Use Cases of PCECC for L3VPN and PWE3

The existing services using MPLS LSP tunnels based on MPLS signalling mechanism such L3VPN, PWE3 and IPv6 can be simplified by using the PCECC to negoitate the label assignments for the L3VPN, PWE3 and Ipv6.

In the case of L3VPN, VPN labels can be negotiated and distributed through the PCECC PCEP among the PE router instead of using the BGP protocols.

Example described in this section is based on network configurations illustrated using the following figure:



Example: Using PCECC for L3VPN and PWE3

In the cast PWE3, instead of using the LDP signalling protocols, the lable and port pairs assigned to each pseudowire can be negotiated through PCECC among the PE rotuers and the corresponding forwarding entries will be distributed into each PE routers through the extended PCEP protocols.

10. Using PCECC for Traffic Classification Information

When a TE-LSP is set up, the head end needs to know:

o how to use it

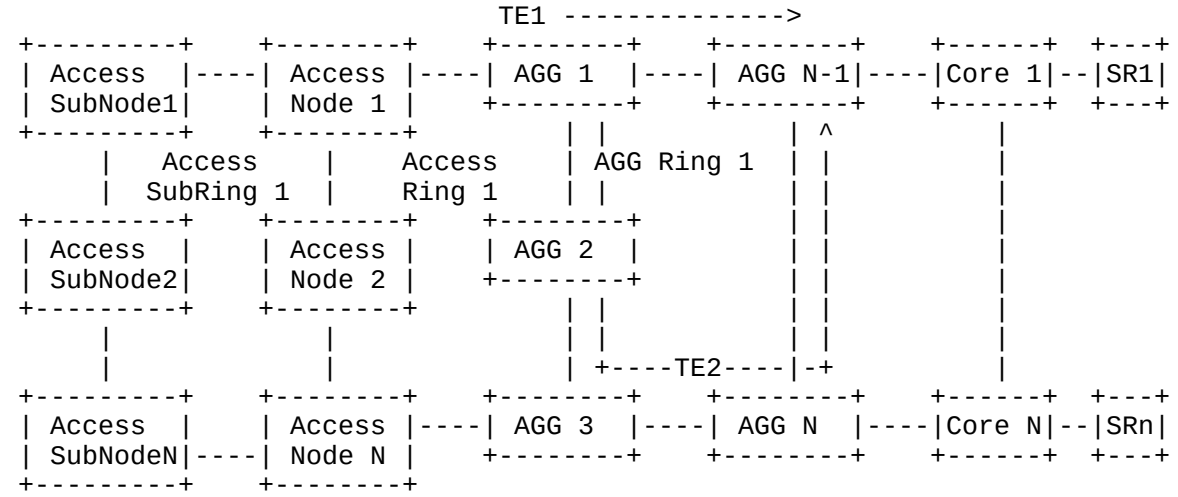
- o What traffic to send on the LSP
- o Whether it is a virtual link
- o Whether to advertise it in the IGP
- o What bits of this information to signal to the tail end

PCEP allows an Active PCE to set up or modify LSPs. But we have no way to tell the head end how to use the LSP. This is because of history. It used to be the LER that made the request of the PCE, so it knew why it wanted the LSP.

With the PCECC architecture by extending the PCEP protocols, it is easy to carry this information such as how to use the LSP, how to advertise the LSP and other extra signaling information.

11. PCECC Load Balancing (LB) Use Case

Very often many service providers use TE tunnels for solving issues with non-deterministic paths in their networks. One example of such applications is usage of TEs in the mobile backhaul (MBH). Let’s consider the following typicall topology.



This MBH architecture uses L2 access rings and subrings. L3 starts at aggregation. For the sake of simplicity here we have only one access subring,access ring and aggregation ring (AGG1...AGGN), connected by Nx10GE interfaces. Aggregation domain runs its own IGP. There are two Egress routers (AGG N-1,AGG N) that are connected to the Core domain via L2 interfaces. Core also have connections to service routers,

RSVP TEs are used for MPLS transport inside the ring. There could be at least 2 tunnels (one way) from each AGG router to egress AGG routers. There are also many L2 access rings connected to AGG routers.

Service deployment made by means of either L2VPNs (VPLS) or L3VPNs. Those services use MPLS TE as transport towards egress AGG routers. TE tunnels could be also used as transport towards service routers in case of seamless MPLS based architecture in the future.

There is a need to solve the following tasks:
o Perform automatic LB amongst TE tunnels according to current traffic load
o TE bandwidth (BW) management: Provide guaranteed BW for specific service: HSI,IPTV, etc., provide time-based BW reservation (BoD)
o Simplify development of TE tunnels (go away from manual provisioning)
o Provide flexibility for Service Router placement (anywhere in the network by creation of transport LSPs to them)

Since other tasks are considered in other PCECC use cases above, hereafter we will focus only on load balancing (LB) task. LB task could be solved by means of PCECC in the following way:

o After application or network service or operator will ask SDN controller (PCECC) for LSP based LB between AGG X and AGG N/AGG N-1 (egress AGG routers which have connections to core) via North Bound Interface (NBI such as REST API), PCECC SHOULD ask for constrains for that particular calculation (i.e. LSP type: traditional CR-LSP or SR-TE LSP, bandwidth, inclusion or exclusion specific links or nodes, number of paths, shortest path or minimum cost tree, need for disjoint LSP paths etc.).
o PCECC MUST calculate N P2P LSPs according to given constrains, calculation is based on results of Objective Function (OF), that includes same source and destination routers IDs, same or different bandwidth (BW) , different links (in case of disjoint paths) and other constrains from Step 1.
o Depending on given LSP type (CR-LSP or SR-TE), PCECC SHOULD create different labels (aka different label spaces, it MAY also require label space negotiation procedure between PCECC and PCCs) for calculated LSPs from egress nodes AGG N-1 and AGG N towards ingress AGG X node.
o PCECC SHOULD send PCInitiate PCEP message [I-D.crabbe-pce-pce-initiated-lsp] towards ingress AGG X router(PCC) for each of N LSPs and receives PCRpt PCEP message [I-D.ietf-pce-stateful-pce] back from him.
o If LSP type is CR-LSP, PCECC MUST send PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce-controller] PCEP message to each node along the path with label information for each of N LSPs. If LSP type is SR-TE, PCECC also MUST send PCLabelUpd PCEP message

to each node along the path with label information (Node-ID and Adjacency-ID segment (label) list) specific to that node. Then PCECC SHOULD send PCUpd PCEP message to the ingress AGG X router with information about new LSP and AGG X(PCC) SHOULD send PCEP PCRpt back with LSP status:Up.
o Now each router along the LSP has corresponding label forwarding state for each of N LSPs.
o AGG X as ingress router now have N LSPs towards AGG N and AGG N-1 which are available for installing to router's RIB and LB of traffic between them. Traffic distribution between those LSPs depends on particular realization of hash-function on that router.
o Since PCECC MUST know as LSDB as TEDB (TE state) he can manage and prevent possible oversubscriptions and limit number of available LB states.

12. Using reliable P2MP TE based multicast delivery for distributed computations (MapReduce-Hadoop)

MapReduce model of distributed computations in computing clusters is widely deployed. In Hadoop 1.0 architecture MapReduce operations on big data performs by means of Master-Slave architecture in the Hadoop Distributed File System (HDFS),where NameNode has the knowledge about resources of the cluster and where actual data (chunks) for particular task are located (which DataNode). Each chunk of data (64MB or more) should have 3 saved copies in different DataNodes based on their proximity.

Proximity level currently has semi-manual allocation and based on Rack IDs (Assumption is that closer data are better because of access speed/smaller latency).

JobTracker node is responsible for computation tasks, scheduling across DataNodes and also have Rack-awareness. Currently transport protocols between NameNode/JobTracker and DataNodes are based on IP unicast. It has simplicity as pros but has numerous drawbacks related with its flat approach.

It is clear that we should go beyond of one DC for Hadoop cluster creation and move towards distributed clusters. In that case we need to handle performance and latency issues.
Latency depends on speed of light in fiber links and also latency introduced by intermediate devices in between. The last one is closely correlated with network device architecture and performance. Current performance of NPU based routers should be enough for creating distribute Hadoop clusters with predicted latency. Performance of SW based routers (mainly as VNF) together with additional HW features such as DPDK are promising but require additional research and testing.

Main question is how can we create simple but effective architecture for distributed Hadoop cluster?

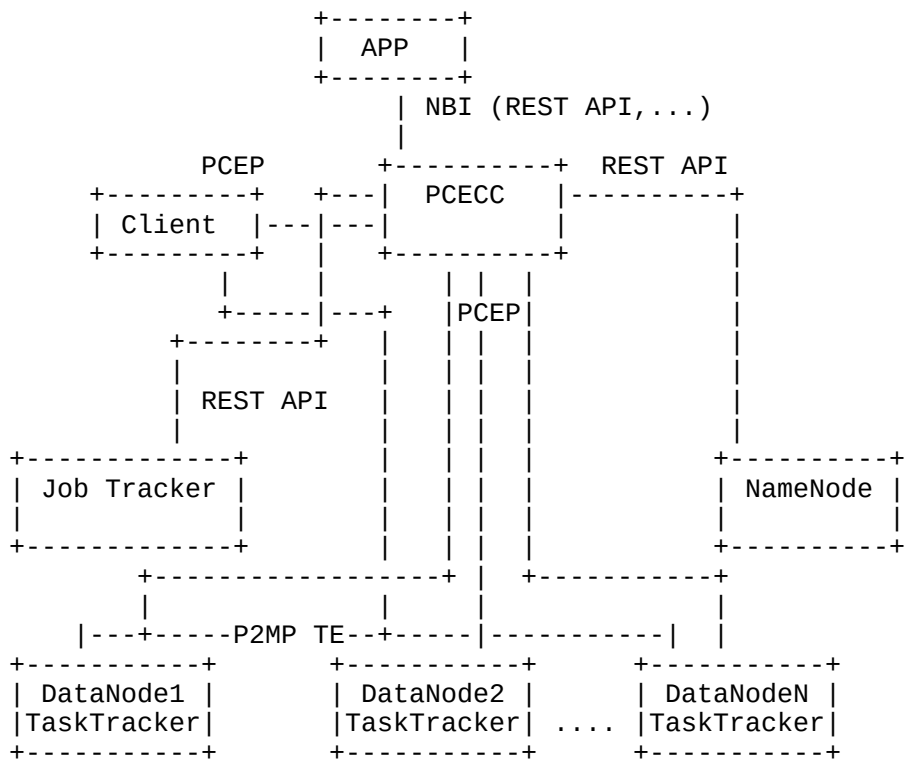
There are number of researches [Multicast Tree Map-Reduce...] which show how usage of multicast tree could improve speed of resource or cluster members discovery inside the cluster as well as increase redundancy in communications between cluster nodes.

Is traditional IP based multicast enough for that? We doubt it because it requires additional control plane (IGMP, PIM) and a lot of signaling, that is not suitable for high performance computations, that are very sensitive to latency.

P2MP TE tunnels looks much more suitable as potential solution for creation of multicast based communications between Master and Slave nodes inside cluster. Obviously these P2MP tunnels should be dynamically created and turned down (no manual intervention). Here is there PCECC comes to play. His main task is to create optimal topology of each partucular request for MapReduce computation and also create P2MP tunnels with needed parameters such as badnwidth and delay.

This solution would require to use MPLS label based forwarding inside the cluster. Usage of label based forwarding inside DC was proposed by Yandex [MPLS in DC...] Technically it is already possible because mpls on switches is already supported by some vendors, mpls aslo exists on Linux and OVS.

The following framework can make this task:



Communication between Master nodes (JobTracker and NameNode) and PCECC via REST API MAY be either done directly or via cluster manager such as Mesos.

Phase 1: Distributed cluster resources discovery
During this phase Master Nodes SHOULD identify and find available Slave nodes according to computing request from application (APP). NameNode SHOULD query PCECC about available DataNodes, NameNode MAY provide additional constrains to PCECC such as topological proximity, redundancy level.

PCECC SHOULD analyze the topology of distributed cluster and perform constrain based path calculation [RFC7334] from client towards most suitable NameNodes. PCECC SHOULD reply to NameNode the list of most suitable DataNodes and their resource capabilities. Topology discovery mechanism for PCECC will be added later to that framework.

Phase 2: PCECC SHOULD create P2MP LSP from client towards those DataNodes by means of PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce -controller] PCEP messages following previously calculated path.

Phase 3. NameNode SHOULD send this information to client, PCECC informs client about optimal P2MP path towards DataNodes via PCEP PCUpd message.

Phase 4. Client sends data blocks to those DataNodes for writing via created P2MP tunnel.

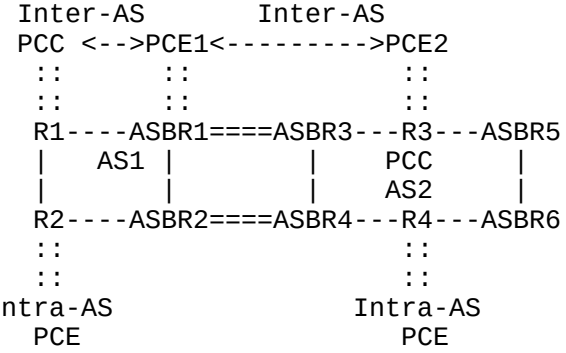
When this task will be finished, P2MP tunnel MAY be turned down.

13. PCECC and Inter-AS TE

There are three signalling options for establishing Inter-AS TE LSP: contiguous TE LSP [RFC5151], stitched inter-AS TE LSP [RFC5150], nested TE LSP [RFC4206].

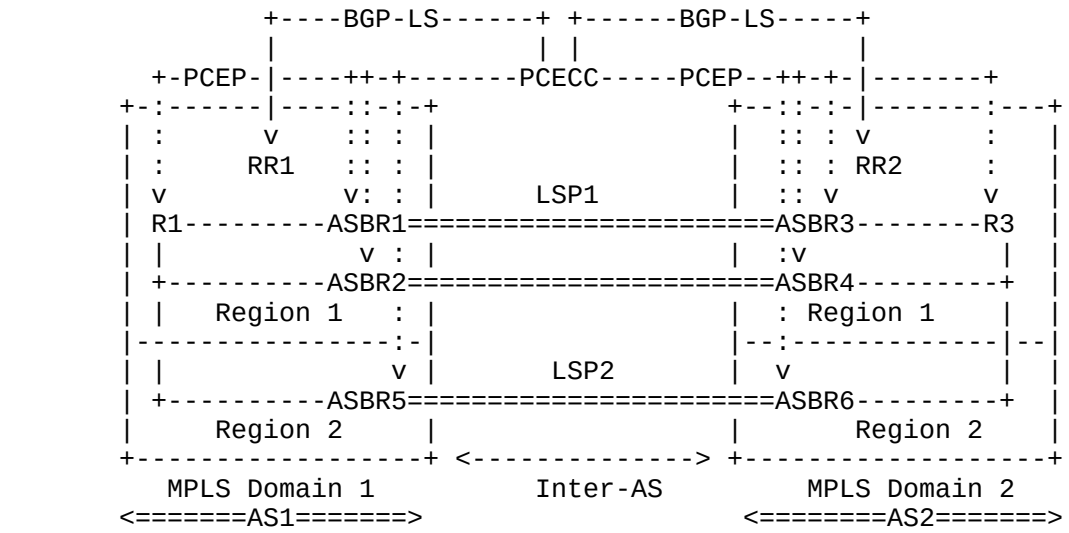
Requirements for PCE-based Inter-AS setup [RFC5376] describe the approach and PCEP fucntionality that are needed for establishing Inter-AS TE LSPs.

[RFC5376] also gives Inter- and Intra-AS PCE Reference Model that is provided below in shorten form for the sake of simplicity.



Shorten form of Inter- and Intra-AS PCE Reference Model [RFC5376]

Hereatfter we will discuss a simplified Inter-AS case when both AS1 and AS2 belong to the same service provider administration. In that case Inter and Intra-AS PCEs could be combined in one single PCE if such combined PCE performance is enough for handling all Path Computation Requests. Even more in that particular case we potentially could use single PCE for both ASes if his scalability and performance are enough, we just will need interfaces (PCEP and BGP-LS) to both domains. SDN controller’s redundancy mechanisms are out of scope in our case. Thus routers in AS1 and AS2 (PCCs) will send Path Computation Requests towards same PCE.



Particular case of Inter-AS PCE Reference Model

In one particular case of PCECC Inter-AS TE scenario service provider controls both domains (AS1 and AS2), each of them have own IGP and MPLS transport. The need is to setup Inter-AS LSPs for transporting different services on top of them (Voice,L3 VPN etc.) Inter-AS links with different capacity exist in several regions. The task is not only to provision those Inter-AS LSPs with given constrains but also calculate the path and pre-setup the backup Inter-AS LSPs that will be used if main LSP fails.

For the figure above it would be that LSP1 from R1 to R3 SHOULD go via ASBR1 and ASBR3, and it is the main Inter-AS LSP. R1-R3 LSP2 that SHOULD go via ASBR5 and ASBR6 is the backup one. Depending on Inter-AS TE type, backup LSP could be used either by head-end R1 or ASBR1.

After the addition of PCECC functionality to PCE (SDN controller), PCECC based Inter-AS TE model SHOULD follow as PCECC usecase for TE LSP (case 6 above) as requirements of [RFC5376] with the following details:

o Since PCECC MUST know the topology of both domains AS1 and AS2, PCECC MUST establish BGP-LS peering with routers (or RRs) in both domains

o PCECC MUST have SBI (PCEP) connectivity towards all routers in both domains (see also section 4 in [RFC5376])

o After operator’s application or service orchetsrator will create request for topology of specific service, PCECC SHOULD receive that request via NBI (NBI type is implementation dependent, MAY be NETCONF/Yang, REST etc.). Then PCECC SHOULD calculate Objective Function (OF) for optimal path with given constrains (i.e. LSP type, bandwidth etc.), including those from [RFC5376]: priority, AS sequence, preffered ASBR, disjoint paths, protection. On this step we would have two paths: R1-ASBR1-ASBR3-R3, R1-ASBR5-ASBR6-R3

o Depending on given LSP type (CR-LSP or SR-TE), PCECC SHOULD create different labels (aka different label spaces, it MAY also require label space negotiation procedure between PCECC and PCCs) for calculated LSPs from egress node in one AS towards ingress in another AS.

o PCECC SHOULD send PCInitiate PCEP message [I-D.crabbe-pce-pce-initiated-lsp] towards ingress router R1 (PCC) in AS1 and receive PCRpt PCEP message [I-D.ietf-pce-stateful-pce] back from him.

o If LSP type is CR-LSP, PCECC MUST send PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce-controller] PCEP message to each node along the path (ASBR1-ASBR3-R3, ASBR5-ASBR6-R3) in both ASes with label information for that LSP.
If LSP type is SR-TE, PCECC also MUST send PCLabelUpd PCEP message to each node along the path in aboth Ases with label information (Node-ID and Adjacency-ID segment (label) list) specific to that node.

o Then PCECC SHOULD send PCUpd PCEP message to the ingress router R1 in AS1 with information about new LSP and the R1 router SHOULD send PCEP PCRpt back

with LSP1 and LSP2 status:Up.

o After that step R1 SHOULD have main and backup TEs (LSP1 and LSP2) towards R3 up. It is up to implementation how to put this TEs to R1's RIB and how to make switchover to backup LSP2 if LSP1 fails.

14. The Considerations for PCECC Procedure and PCEP extensions

The PCECC's procedures and PCEP extensions is defined in [I-D.zhao-pce-pcep-extension-for-pce-controller].

15. IANA Considerations

This document does not require any action from IANA.

16. Security Considerations

TBD.

17. Acknowledgments

We would like to thank Robert Tao, Changjiang Yan, Tieying Huang, Adrian Farrel, Sergio Belotti and Dieter Beller, Andrey Elperin and Evgeniy Brodskiy for their useful comments and suggestions.

18. References

18.1. 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>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <http://www.rfc-editor.org/info/rfc5440>.

18.2. Informative References

[RFC5441] Vasseur, JP., Ed., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, DOI 10.17487/RFC5441, April 2009, <http://www.rfc-editor.org/info/rfc5441>.

[RFC5541] Le Roux, JL., Vasseur, JP., and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, DOI 10.17487/RFC5541, June 2009, <http://www.rfc-editor.org/info/rfc5541>.

[RFC5376] N. Bitar, R. Zhang, K. Kumaki "Inter-AS Requirements for the Path Computation Element Communication Protocol (PCECP)", RFC 5376, DOI 10.17487/RFC5376, November 2008 <http://www.rfc-editor.org/info/rfc5376>.

[I-D.filsfils-spring-segment-routing] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.

[I-D.ietf-pce-stateful-pce] Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-14 (work in progress), May 2016.

[I-D.crabbe-pce-pce-initiated-lsp] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-crabbe-pce-pce-initiated-lsp-05 (work in progress), October 2015.

[I-D.ali-pce-remote-initiated-gmpls-lsp]

Ali, Z., Sivabalan, S., Filsfils, C., Varga, R., Lopez, V., Dios, O., and X. Zhang, "Path Computation Element Communication Protocol (PCEP) Extensions for remote-initiated GMPLS LSP Setup", draft-ali-pce-remote-initiated-gmpls-lsp-03 (work in progress), February 2014.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-06 (work in progress), December 2015.

[I-D.psenak-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-psenak-ospf-segment-routing-extensions-05 (work in progress), June 2014.

[I-D.sivabalan-pce-segment-routing]
Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., Raszuk, R., Lopez, V., and J. Tantsura, "PCEP Extensions for Segment Routing", draft-sivabalan-pce-segment-routing-03 (work in progress), July 2014.

[I-D.li-mpls-global-label-usecases]
Li, Z., Zhao, Q., Yang, T., Raszuk, R., and L. Fang, "Usecases of MPLS Global Label", draft-li-mpls-global-label-usecases-03 (work in progress), October 2015.

[I-D.li-mpls-global-label-framework]
Li, Z., Zhao, Q., Chen, X., Yang, T., and R. Raszuk, "A Framework of MPLS Global Label", draft-li-mpls-global-label-framework-02 (work in progress), July 2014.

[I-D.zhao-pce-pcep-extension-for-pce-controller]
Zhao, Q., Li, Z., Dhody, D., and C. Zhou, "PCEP Procedures and Protocol Extensions for Using PCE as a Central Controller (PCECC) of LSPs", draft-zhao-pce-pcep-extension-for-pce-controller-03 (work in progress), March 2016.

[I-D.ietf-spring-resiliency-use-cases]
Francois, P., Filsfils, C., Decraene, B., and R. Shakir, "Use-cases for Resiliency in SPRING", draft-ietf-spring-resiliency-use-cases-02 (work in progress), December 2015.

[MPLS in DC...]
Afanasiev, D., Ginsburg, D., "MPLS in DC and inter-DC networks: the unified forwarding mechanism for network programmability at scale "

[Multicast Tree Map-Reduce...]
Lee, Kyungyong., Dr. Boykin, P. Oscar., Dr.Figueiredo, Renato J., "Multicast Tree Map-Reduce: Self-organizing Resource Discovery and Monitoring using Structured P2P Systems"

Authors' Addresses

Quintin Zhao
Huawei Technologies
125 Nagog Technology Park
Acton, MA 01719
US

EMail: quintin.zhao@huawei.com

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

EMail: lizhenbin@huawei.com

Boris Khasanov
Huawei Technologies
Moskovskiy Prospekt 97A
St.Petersburg 196084
Russia

EMail: khasanov.boris@huawei.com

King Ke
Tencent Holdings Ltd.
Shenzhen
China

EMail: kinghe@tencent.com

Luyuan Fang
Microsoft

EMail: lufang@microsoft.com

Chao Zhou
Cisco Systems

EMail: chao.zhou@cisco.com

Boris Zhang
Telus Communications

EMail: Boris.zhang@telus.com

Artem Rachitskiy
Mobile TeleSystems JLLC
Nezavisimosti ave., 95
Minsk 220043
Belarus

EMail: arachitskiy@mts.by

Anton Gulida
Mobile TeleSystems JLLC
Nezavisimosti ave., 95
Minsk 220043
Belarus

EMail: agulida@mts.by

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 13, 2019

L. Zheng
G. Zheng
Huawei Technologies
G. Mirsky
ZTE Corp.
R. Rahman
F. Iqbal
Cisco Systems

January 9, 2019

YANG Data Model for LSP-Ping
draft-zheng-mpls-lsp-ping-yang-cfg-10

Abstract

When an LSP fails to deliver user traffic, the failure cannot always be detected by the MPLS control plane. RFC 8029 defines a mechanism that would enable users to detect such failure and to isolate faults. YANG, defined in RFC 6020 and RFC 7950, is a data modeling language used to specify the contents of a conceptual data stores that allows networked devices to be managed using NETCONF, as specified in RFC 6241. This document defines a YANG data model that can be used to configure and manage LSP-Ping.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on July 13, 2019.

Table of Contents

1. Introduction

When an LSP fails to deliver user traffic, the failure cannot always be detected by the MPLS control plane. [RFC8029] defines a mechanism that would enable users to detect such failure and to isolate faults. YANG, defined in [RFC6020] and [RFC7950], is a data modeling language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. This document defines a YANG data model that can be used to configure and manage LSP-Ping [RFC8029].

The rest of this document is organized as follows. Section 2 presents the scope of this document. Section 3 provides the design of the LSP-Ping configuration data model in details by containers. Section 4 presents the complete data hierarchy of LSP-Ping YANG model. Section 5 discusses the interaction between LSP-Ping data model and other MPLS tools data models. Section 6 specifies the YANG module and section 7 lists examples which conform to the YANG module specified in this document. Finally, security considerations are discussed in Section 8.

This version of the LSP Ping data model conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

1.1. Requirements Language

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

1.2. Support of Long Running Command with NETCONF

LSP Ping is one of the examples of what can be described as "long-running operation". Unlike most of the configuration operations that result in single response execution of an LSP Ping triggers multiple responses from a node under control. The question of implementing the long-running operation in NETCONF is still open and possible solutions being discussed:

1. Consecutive Remote Processing Calls (RPC) to poll for results.

2. Model presented in [RFC4560].

3. The one outlined in [I-D.mahesh-netconf-persistent].

The problem of long-running operation as well can be considered as a case of controlling and obtaining results from a Measurement Agent (MA) as defined in [RFC7594].

2. Scope

The fundamental mechanism of LSP-Ping is defined in [RFC8029]. Extensions of LSP-Ping has been developed over the years. There are extensions for performing LSP ping, for example, over P2MP MPLS LSPs [RFC6425] or for Segment Routing IGP Prefix and Adjacency SIDs with an MPLS data plane [RFC8287]. These extensions will be considered in a later update of this document.

3. Design of the Data Model

This YANG data model is defined to be used to configure and manage LSP-Ping and it provides the following features:

1. The configuration of control information of an LSP-Ping test.

2. The configuration of schedule parameters of an LSP-Ping test.

3. Display of result information of an LSP-Ping test.

The top-level container lsp-pings holds the configuration of the control information, schedule parameters and result information for multiple instances of LSP-Ping test.

3.1. The Configuration of Control Information

Container lsp-pings:lsp-ping:control-parameters defines the configuration parameters which control an LSP-Ping test. Examples are the target-fec-type/target-fec of the echo request packet and the reply mode of the echo reply packet. Values of some parameters may be auto-assigned by the system, but in several cases, there is a requirement for configuration of these parameters. Examples of such parameters are source address and outgoing interface.

The data hierarchy for control information configuration is presented below:

```
module: ietf-lsp-ping
   +--rw lsp-pings
      +--rw lsp-ping* [lsp-ping-name]
         +--rw lsp-ping-name          string
         +--rw control-parameters
         |  +--rw target-fec-type?       target-fec-type
         |  +--rw (target-fec)?
         |  |  +--:(ip-prefix)
         |  |  |  +--rw ip-address?            inet:ip-address
         |  |  +--:(bgp)
         |  |  |  +--rw bgp?                   inet:ip-address
         |  |  +--:(rsvp)
         |  |  |  +--rw tunnel-interface?      string
         |  |  +--:(vpn)
         |  |  |  +--rw vrf-name?              uint32
         |  |  |  +--rw vpn-ip-address?        inet:ip-address
         |  |  +--:(pw)
         |  |  |  +--rw vcid?                  uint32
         |  |  +--:(vpls)
         |  |     +--rw vsi-name?              string
         |  +--rw traffic-class?         uint8
         |  +--rw reply-mode?            reply-mode
         |  +--rw timeout?               uint32
         |  +--rw timeout-units?         units
         |  +--rw interval?              uint32
         |  +--rw interval-units?        units
         |  +--rw probe-count?           uint32
         |  +--rw data-size?             uint32
         |  +--rw data-fill?             string
         |  +--rw description?           string
         |  +--rw source-address?        inet:ip-address
         |  +--rw ttl?                   uint8
         |  +--rw (outbound)?
         |     +--:(interface)
         |     |  +--rw interface-name?        string
         |     +--:(nexthop)
         |        +--rw nexthop?               inet:ip-address
```

3.2. The Configuration of Schedule Parameters

Container lsp-pings:lsp-ping:scheduling-parameters defines the schedule parameters of an LSP-Ping test, which describes when to start and when to end the test. Four start modes and three end modes are defined respectively. To be noted that, the configuration of "interval" and "probe-count" parameter defined in container lsp-pings:lsp-ping:control-parameters could also determine when the test ends implicitly. All these three parameters are optional.If the user

does not configure either "interval" or "probe-count" parameter, then the default values will be used by the system. If the user configures "end-test", then the actual end time of the LSP-Ping test is the smaller one between the configuration value of "end-test" and the time implicitly determined by the configuration value of "interval"/"probe-count".

The data hierarchy for schedule information configuration is presented below:

```
module: ietf-lsp-ping
   +--rw lsp-pings
      +--rw lsp-ping* [lsp-ping-name]
         +--rw lsp-ping-name          string
         +--rw control-parameters
         ...
         +--rw scheduling-parameters
         |  +--rw (start-test)?
         |  |  +--:(now)
         |  |  |  +--rw start-test-now?           empty
         |  |  +--:(at)
         |  |  |  +--rw start-test-at?            yang:date-and-time
         |  |  +--:(delay)
         |  |  |  +--rw start-test-delay?         uint32
         |  |  |  +--rw start-test-delay-units?   units
         |  |  +--:(daily)
         |  |     +--rw start-test-daily?         yang:date-and-time
         |  +--rw (end-test)?
         |     +--:(at)
         |     |  +--rw end-test-at?              yang:date-and-time
         |     +--:(delay)
         |     |  +--rw end-test-delay?           uint32
         |     |  +--rw end-test-delay-units?     units
         |     +--:(lifetime)
         |        +--rw end-test-lifetime?        uint32
         |        +--rw lifetime-units?           units
```

3.3. Display of Result Information

Container lsp-pings:lsp-ping:result-info shows the result of the current LSP-Ping test. Both the statistical result e.g. min-rtt, max-rtt, and per test probe result e.g. return code, return subcode, are shown.

The data hierarchy for display of result information is presented below:

```
module: ietf-lsp-ping
   +--rw lsp-pings
      +--rw lsp-ping* [lsp-ping-name]
         +--rw lsp-ping-name          string
         +--rw control-parameters
         ...
         +--rw scheduling-parameters
         ...
         +--ro result-info
            +--ro operational-status?    operational-status
            +--ro source-address?        inet:ip-address
            +--ro target-fec-type?       target-fec-type
            +--ro (target-fec)?
            |  +--:(ip-prefix)
            |  |  +--ro ip-address?            inet:ip-address
            |  +--:(bgp)
            |  |  +--ro bgp?                   inet:ip-address
            |  +--:(rsvp)
            |  |  +--ro tunnel-interface?      string
            |  +--:(vpn)
            |  |  +--ro vrf-name?              uint32
            |  |  +--ro vpn-ip-address?        inet:ip-address
            |  +--:(pw)
            |  |  +--ro vcid?                  uint32
            |  +--:(vpls)
            |     +--ro vsi-name?              string
            +--ro min-rtt?               uint32
            +--ro max-rtt?               uint32
            +--ro average-rtt?           uint32
            +--ro probe-responses?       uint32
            +--ro sent-probes?           uint32
            +--ro sum-of-squares?        uint32
            +--ro last-good-probe?       yang:date-and-time
            +--ro probe-results
               +--ro probe-result* [probe-index]
                  +--ro probe-index        uint32
                  +--ro return-code?       uint8
                  +--ro return-sub-code?   uint8
                  +--ro rtt?               uint32
                  +--ro result-type?       result-type
```

4. Data Hierarchy

The complete data hierarchy of LSP-Ping YANG model is presented below.

```
module: ietf-lsp-ping
```

```
      +--rw lsp-pings
         +--rw lsp-ping* [lsp-ping-name]
            +--rw lsp-ping-name          string
            +--rw control-parameters
            |  +--rw target-fec-type?       target-fec-type
            |  +--rw (target-fec)?
            |  |  +--:(ip-prefix)
            |  |  |  +--rw ip-address?            inet:ip-address
            |  |  +--:(bgp)
            |  |  |  +--rw bgp?                   inet:ip-address
            |  |  +--:(rsvp)
            |  |  |  +--rw tunnel-interface?      string
            |  |  +--:(vpn)
            |  |  |  +--rw vrf-name?              uint32
            |  |  |  +--rw vpn-ip-address?        inet:ip-address
            |  |  +--:(pw)
            |  |  |  +--rw vcid?                  uint32
            |  |  +--:(vpls)
            |  |     +--rw vsi-name?              string
            |  +--rw traffic-class?         uint8
            |  +--rw reply-mode?            reply-mode
            |  +--rw timeout?               uint32
            |  +--rw timeout-units?         units
            |  +--rw interval?              uint32
            |  +--rw interval-units?        units
            |  +--rw probe-count?           uint32
            |  +--rw data-size?             uint32
            |  +--rw data-fill?             string
            |  +--rw description?           string
            |  +--rw source-address?        inet:ip-address
            |  +--rw ttl?                   uint8
            |  +--rw (outbound)?
            |     +--:(interface)
            |     |  +--rw interface-name?        string
            |     +--:(nexthop)
            |        +--rw nexthop?               inet:ip-address
            +--rw scheduling-parameters
            |  +--rw (start-test)?
            |  |  +--:(now)
            |  |  |  +--rw start-test-now?           empty
            |  |  +--:(at)
            |  |  |  +--rw start-test-at?            yang:date-and-time
            |  |  +--:(delay)
            |  |  |  +--rw start-test-delay?         uint32
            |  |  |  +--rw start-test-delay-units?   units
            |  |  +--:(daily)
            |  |     +--rw start-test-daily?         yang:date-and-time
            |  +--rw (end-test)?
```

```
|      +--:(at)
|      |  +--rw end-test-at?              yang:date-and-time
|      +--:(delay)
|      |  +--rw end-test-delay?           uint32
|      |  +--rw end-test-delay-units?     units
|      +--:(lifetime)
|         +--rw end-test-lifetime?        uint32
|         +--rw lifetime-units?           units
|
+--ro result-info
   +--ro operational-status?    operational-status
   +--ro source-address?        inet:ip-address
   +--ro target-fec-type?       target-fec-type
   +--ro (target-fec)?
   |  +--:(ip-prefix)
   |  |  +--ro ip-address?            inet:ip-address
   |  +--:(bgp)
   |  |  +--ro bgp?                   inet:ip-address
   |  +--:(rsvp)
   |  |  +--ro tunnel-interface?      string
   |  +--:(vpn)
   |  |  +--ro vrf-name?              uint32
   |  |  +--ro vpn-ip-address?        inet:ip-address
   |  +--:(pw)
   |  |  +--ro vcid?                  uint32
   |  +--:(vpls)
   |     +--ro vsi-name?              string
   +--ro min-rtt?               uint32
   +--ro max-rtt?               uint32
   +--ro average-rtt?           uint32
   +--ro probe-responses?       uint32
   +--ro sent-probes?           uint32
   +--ro sum-of-squares?        uint32
   +--ro last-good-probe?       yang:date-and-time
   +--ro probe-results
      +--ro probe-result* [probe-index]
         +--ro probe-index        uint32
         +--ro return-code?       uint8
         +--ro return-sub-code?   uint8
         +--ro rtt?               uint32
         +--ro result-type?       result-type
```

5. Interaction with other MPLS OAM Tools Models

TBA

6. LSP-Ping YANG Module

```
<CODE BEGINS> file "ietf-lsp-ping@2018-11-29.yang"
module ietf-lsp-ping {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-lsp-ping";
  //namespace need to be assigned by IANA
  prefix "lsp-ping";

  import ietf-inet-types {
  prefix inet;
  reference "RFC 6991: Common YANG Types.";
  }
  import ietf-yang-types{
  prefix yang;
  reference "RFC 6991: Common YANG Types.";
  }

  organization "IETF Multiprotocol Label Switching Working Group";

    contact
    "WG Web: http://tools.ietf.org/wg/mpls/
     WG List: mpls@ietf.org

     Editor: Greg Mirsky
          gregimirsky@gmail.com
     Editor: Lianshu Zheng
          vero.zheng@huawei.com
     Editor: Guangying Zheng
          zhengguangying@huawei.com
     Editor: Reshad Rahman
          rrahman@cisco.com
     Editor: Faisal Iqbal
          faiqbal@cisco.com";

    description
    "This YANG module specifies a vendor-independent model
     for the LSP Ping.

     This YANG data model is defined to be used to configure and manage
     LSP-Ping and it provides the following features:
     1.  The configuration of control information of an LSP-Ping test.
     2.  The configuration of schedule parameters of an LSP-Ping test.
     3.  Display of result information of an LSP-Ping test.

        Copyright (c) 2018 IETF Trust and the persons identified as
        the document authors.  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.";

    reference "draft-zheng-mpls-lsp-ping-yang-cfg";

    revision "2018-11-29" {
      description
        "10 version, refine the target fec type,
         as per RFC8029 and update Security Considerations section.";
    reference "draft-zheng-mpls-lsp-ping-yang-cfg";
  }

  typedef target-fec-type {
    type enumeration {
      enum ip-prefix {
        value "0";
        description "IPv4/IPv6 prefix";
      }
      enum bgp {
        value "1";
        description "BGP IPv4/IPv6 prefix";
      }
      enum rsvp {
        value "2";
        description "Tunnel interface";
      }
      enum vpn {
        value "3";
        description "VPN IPv4/IPv6 prefix";
      }
      enum pw {
        value "4";
        description "FEC 128 pseudowire IPv4/IPv6";
      }
      enum vpls {
        value "5";
        description "FEC 129 pseudowire IPv4/IPv6";
      }
    }
    description "Target FEC type, as defined in RFC 8029";
  }
```

```
  typedef reply-mode {
    type enumeration {
      enum do-not-reply {
        value "1";
        description "Do not reply";
      }
      enum reply-via-udp {
        value "2";
        description "Reply via an IPv4/IPv6 UDP packet";
      }
      enum reply-via-udp-router-alert {
        value "3";
        description
          "Reply via an IPv4/IPv6 UDP packet with Router Alert";
      }
      enum reply-via-control-channel {
        value "4";
        description
          "Reply via application level control channel";
      }
    }
    description "Reply mode";
  }

  typedef units  {
    type enumeration {
      enum seconds {
        description "Seconds";
      }
      enum milliseconds {
        description "Milliseconds";
      }
      enum microseconds {
        description "Microseconds";
      }
      enum nanoseconds {
        description "Nanoseconds";
      }
    }
    description "Time units";
  }

  typedef operational-status {
    type enumeration {
      enum enabled {
        value "1";
        description "The Test is active";
      }
```

```
      enum disabled {
        value "2";
        description "The test has stopped";
      }
      enum completed {
        value "3";
        description "The test is completed";
      }
    }
    description "Operational state of an LSP Ping test";
  }

  typedef result-type {
    type enumeration {
      enum success {
        value "1";
        description "The test probe is successful";
      }
      enum fail {
        value "2";
        description "The test probe has failed";
      }
      enum timeout {
        value "3";
        description "The time of the test probe has expired";
      }
    }
    description "Result of each LSP Ping test probe";
  }

  container lsp-pings {
    description "Multi-instance of the LSP Ping test";
    list lsp-ping {
      key "lsp-ping-name";
      description "LSP Ping test";
      leaf lsp-ping-name {
        type string {
            length "1..31";
        }
        mandatory "true";
        description "LSP Ping test name";
      }
      container control-parameters {
        description "Control information of the LSP Ping test";
        leaf target-fec-type {
          type target-fec-type;
          description "Specifies the address type of the Target FEC";
        }
```

```
        choice target-fec {
          case ip-prefix {
            leaf ip-address {
              type inet:ip-address;
              description "IPv4/IPv6 Prefix";
            }
          }
          case bgp {
            leaf bgp {
              type inet:ip-address;
              description "BGP IPv4/IPv6 Prefix";
            }
          }
          case rsvp {
            leaf tunnel-interface {
              type string;
              description "Tunnel interface";
            }
          }
          case vpn {
            leaf vrf-name {
              type uint32;
              description "Layer3 VPN Name";
            }
            leaf vpn-ip-address {
              type inet:ip-address;
              description "Layer3 VPN IPv4 Prefix";
            }
          }
          case pw {
            leaf vcid {
              type uint32;
              description "VC ID";
            }
          }
          case vpls {
            leaf vsi-name {
              type string;
              description "VPLS VSI";
            }
          }
          description "Specifies the type of the Target FEC";
        }
        leaf traffic-class {
          type uint8;
          description "Specifies the Traffic Class";
        }
        leaf reply-mode {
```

```
          type reply-mode;
          description "Specifies the Reply Mode";
        }
        leaf timeout {
          type uint32;
          description
            "Specifies the time-out value for a LSP Ping operation.";
        }
        leaf timeout-units {
          type units;
          description "Time-out units";
        }
        leaf interval {
          type uint32;
          default 1;
          description
            "Specifies the interval between transmissions
             of LSP Ping echo request packets (probes)
             as part of the LSP Ping test.";
        }
        leaf interval-units {
          type units;
          default seconds;
          description "Interval units";
        }
        leaf probe-count {
          type uint32;
          default 5;
          description
            "Specifies the number of probes sent in the LSP Ping test.";
        }
        leaf data-size {
          type uint32;
          description
            "Specifies the size of the data portion to
             be transmitted in an LSP Ping operation, in octets.";
        }
        leaf data-fill {
          type string{
              length "0..1564";
          }
          description
            "Used together with the corresponding
             data-size value to determine how to fill the data
             portion of a probe packet.";
        }
        leaf description {
          type string{
```

```
            length "1..31";
          }
          description "A descriptive name of the LSP Ping test";
        }
        leaf source-address {
          type inet:ip-address;
          description "Specifies the source address";
        }
        leaf ttl {
          type uint8;
          default 255;
          description "Time to live";
        }
        choice outbound {
          case interface {
            leaf interface-name{
              type string{
                length "1..255";
              }
              description "Specifies the outgoing interface";
            }
          }
          case nexthop{
            leaf nexthop {
              type inet:ip-address;
              description "Specifies the nexthop";
            }
          }
          description "Specifies the out interface or nexthop";
        }
      }

      container scheduling-parameters {
        description "LSP Ping test schedule parameter";
        choice start-test{
          case now {
            leaf start-test-now {
              type empty;
              description "Start test now";
            }
          }
          case at {
            leaf start-test-at {
              type yang:date-and-time;
              description "Start test at a specific time";
            }
          }
          case delay {
```

```
            leaf start-test-delay {
              type uint32;
              description "Start after a specific delay";
            }
            leaf start-test-delay-units {
              type units;
              default seconds;
              description "Delay units";
            }
          }
          case daily {
            leaf start-test-daily {
              type yang:date-and-time;
              description "Start test daily";
            }
          }
          description
            "Specifies when the test begins to start,
             include 4 schedule method: start now(1), start at(2),
             start delay(3), start daily(4).";
        }

        choice end-test{
          case at {
            leaf end-test-at{
              type yang:date-and-time;
              description "End test at a specific time";
            }
          }
          case delay {
            leaf end-test-delay {
              type uint32;
              description "End after a specific delay";
            }
            leaf end-test-delay-units {
              type units;
              default seconds;
              description "Delay units";
            }
          }
          case lifetime {
            leaf end-test-lifetime {
              type uint32;
              description "Set the test lifetime";
            }
            leaf lifetime-units {
              type units;
              default seconds;
```

```
            description "Lifetime units";
          }
        }
        description
          "Specifies when the test ends, include 3
           schedule method: end at(1), end delay(2),
           end lifetime(3).";
      }
    }

    container result-info {
      config "false";
      description "LSP Ping test result information";
      leaf operational-status {
        type operational-status;
        description "Operational state of a LSP Ping test";
      }
      leaf source-address {
        type inet:ip-address;
        description "The source address of the test";
      }
      leaf target-fec-type {
        type target-fec-type;
        description "The Target FEC address type";
      }
      choice target-fec {
        case ip-prefix {
          leaf ip-address {
            type inet:ip-address;
            description "IPv4/IPv6 Prefix";
          }
        }
        case bgp {
          leaf bgp {
            type inet:ip-address;
            description "BGP IPv4/IPv6 Prefix";
          }
        }
        case rsvp {
          leaf tunnel-interface {
            type string;
            description "Tunnel interface";
          }
        }
        case vpn {
          leaf vrf-name {
            type uint32;
            description "Layer3 VPN Name";
```

```
            }
            leaf vpn-ip-address {
              type inet:ip-address;
              description "Layer3 VPN IPv4 Prefix";
            }
          }
          case pw {
            leaf vcid {
              type uint32;
              description "VC ID";
            }
          }
          case vpls {
            leaf vsi-name {
              type string;
              description "VPLS VSI";
            }
          }
          description "The Target FEC address";
        }
        leaf min-rtt {
          type uint32;
          description
            "The minimum LSP Ping round-trip-time (RTT)
             received measured in usec.";
        }
        leaf max-rtt {
          type uint32;
          description
            "The maximum LSP Ping round-trip-time (RTT)
             received measured in usec.";
        }
        leaf average-rtt {
          type uint32;
          description
            "The current average LSP Ping round-trip-time
             (RTT) measured in usec.";
        }
        leaf probe-responses {
          type uint32;
          description
            "Number of responses received for the
             corresponding LSP Ping test.";
        }
        leaf sent-probes {
          type uint32;
          description
            "Number of probes sent for the
```

```
           corresponding LSP Ping test.";
        }
        leaf sum-of-squares {
          type uint32;
          description
            "The sum of the squares of RTT,
             calculated as the sum of the squared
             differences between each RTT and the overall
             mean RTT, for all replies received.";
        }
        leaf last-good-probe {
          type yang:date-and-time;
          description
            "Date and time when the last response
             was received for a probe.";
        }

        container probe-results {
          description "Result info of test probes";
          list probe-result {
            key "probe-index";
            description "Result info of each test probe";
            leaf probe-index {
              type uint32;
              config false;
              description "Probe index";
            }
            leaf return-code {
              type uint8;
              config false;
              description "The Return Code set in the echo reply";
            }
            leaf return-sub-code {
               type uint8;
               config false;
               description
                  "The Return Sub-code set in the echo reply.";
            }
            leaf rtt {
              type uint32;
              config false;
              description "The round-trip-time (RTT) received";
            }
            leaf result-type {
              type result-type;
              config false;
              description "The probe result type";
            }
```

```
          }
        }
      }
    }
  }
}
<CODE ENDS>
```

7. Examples

The following examples show the netconf RPC communication between client and server for one LSP-Ping test case.

7.1. Configuration of Control Information

Configure the control-parameters for sample-test-case.

```
Request from netconf client:
<rpc
  message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <lsp-pings xmlns="urn:ietf:params:xml:ns:yang:ietf-lsp-ping">
        <lsp-ping>
          <lsp-ping-name>sample-test-case</lsp-ping-name>
          <control-parameters>
            <target-fec-type>ip-prefix</target-fec-type>
            <ip-prefix>2001:db8::1:100/64</ip-prefix>
            <reply-mode>reply-via-udp</reply-mode>
            <timeout>1</timeout>
            <timeout-units>seconds</timeout-units>
            <interval>1</interval>
            <interval-units>seconds</interval-units>
            <probe-count>6</probe-count>
            <admin-status>enabled</admin-status>
            <data-size>64</data-size>
            <data-fill>this is a lsp ping test</data-fill>
            <source-address>2001:db8::4</source-address>
            <ttl>56</ttl>
          </control-parameters>
        </lsp-ping>
      </lsp-pings>
    </config>
  </edit-config>
</rpc>

Reply from netconf server:
<rpc-reply
  message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

7.2. The Configuration of Schedule Parameters

Set the scheduling-parameters for sample-test-case to start the test.

```
Request from netconf client:
<rpc
  message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <lsp-pings xmlns="urn:ietf:params:xml:ns:yang:ietf-lsp-ping">
        <lsp-ping>
          <lsp-ping-name>sample-test-case</lsp-ping-name>
          <scheduling-parameters>
            <start-test-now/>
          </scheduling-parameters>
        </lsp-ping>
      </lsp-pings>
    </config>
  </edit-config>
</rpc>

Reply from netconf server:
<rpc-reply
  message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

7.3. Display of Result Information

```
Get the result-info of sample-test-case.

Request from netconf client:
<rpc
  message-id="103" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <lsp-pings xmlns="urn:ietf:params:xml:ns:yang:ietf-lsp-ping">
       <lsp-ping>
         <lsp-ping-name>sample-test-case</lsp-ping-name>
           <result-info/>
       </lsp-ping>
      </lsp-pings>
    </filter>
    </get>
</rpc>

Reply from netconf server:
<rpc-reply
  message-id="103" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
<data>
  <lsp-pings xmlns="urn:ietf:params:xml:ns:yang:ietf-lsp-ping">
    <lsp-ping>
      <lsp-ping-name>sample-test-case</lsp-ping-name>
      <result-info>
        <operational-status>completed</operational-status>
        <source-address>2001:db8::4</source-address>
        <target-fec-type>ip-prefix</target-fec-type>
        <ip-prefix>2001:db8::1:100/64</ip-prefix>
        <min-rtt>10</min-rtt>
        <max-rtt>56</max-rtt>
        <average-rtt>36</average-rtt>
        <probe-responses>6</probe-responses>
        <sent-probes>6</sent-probes>
        <sum-of-squares>8882</sum-of-squares>
        <last-good-probe>2015-07-01T10:36:56<last-good-probe>
        <probe-results>
          <probe-result>
            <probe-index>0</probe-index>
            <return-code>0</return-code>
            <return-sub-code>3</return-sub-code>
            <rtt>10</rtt>
            <result-type>success</result-type>
          </probe-result>
          <probe-result>
            <probe-index>1</probe-index>
            <return-code>0</return-code>
            <return-sub-code>3</return-sub-code>
            <rtt>56</rtt>
            <result-type>success</result-type>
          </probe-result>
          <probe-result>
            <probe-index>2</probe-index>
            <return-code>0</return-code>
            <return-sub-code>3</return-sub-code>
            <rtt>35</rtt>
            <result-type>success</result-type>
          </probe-result>
          <probe-result>
            <probe-index>3</probe-index>
            <return-code>0</return-code>
            <return-sub-code>3</return-sub-code>
            <rtt>38</rtt>
            <result-type>success</result-type>
          </probe-result>
          <probe-result>
            <probe-index>4</probe-index>
            <return-code>0</return-code>
```

```
                <return-sub-code>3</return-sub-code>
                <rtt>36</rtt>
                <result-type>success</result-type>
              </probe-result>
              <probe-result>
                <probe-index>5</probe-index>
                <return-code>0</return-code>
                <return-sub-code>3</return-sub-code>
                <rtt>41</rtt>
                <result-type>success</result-type>
              </probe-result>
            </probe-results>
          </result-info>
        </lsp-ping>
      </lsp-pings>
    </data>
  </rpc-reply>
```

8. Security Considerations

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

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a pre-configured 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 an adverse effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

TBD

Unauthorized access to any data node of these subtrees can adversely affect the routing subsystem of both the local device and the network. This may lead to corruption of the measurement that may result in false corrective action, e.g., false negative or false positive. That could be, for example, prolonged and undetected

deterioration of the quality of service or actions to improve the quality unwarranted by the real network conditions.

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:

TBD

Unauthorized access to any data node of these subtrees can disclose the operational state information of VRRP on this device.

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

TBD

The LSP ping YANG module inherits all security consideration of [RFC8029].

9. IANA Considerations

The IANA is requested to as assign a new namespace URI from the IETF XML registry.

URI:TBA

Contributors

Yanfeng Zhang

Huawei Technologies

zhangyanfeng@huawei.com

Sam Aldrin

Google

aldrin.ietf@gmail.com

Acknowledgments

TBD

12. References

12.1. 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>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[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>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

[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>.

12.2. Informative References

[I-D.mahesh-netconf-persistent] Jethanandani, M., "NETCONF and persistent responses", draft-mahesh-netconf-persistent-00 (work in progress), October 2014.

[RFC4560] Quittek, J., Ed. and K. White, Ed., "Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations", RFC 4560, DOI 10.17487/RFC4560, June 2006, <https://www.rfc-editor.org/info/rfc4560>.

[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>.

[RFC6425] Saxena, S., Ed., Swallow, G., Ali, Z., Farrel, A., Yasukawa, S., and T. Nadeau, "Detecting Data-Plane Failures in Point-to-Multipoint MPLS - Extensions to LSP Ping", RFC 6425, DOI 10.17487/RFC6425, November 2011, <https://www.rfc-editor.org/info/rfc6425>.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

[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>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

[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>.

[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>.

Authors' Addresses

Lianshu Zheng
Huawei Technologies
China

Email: vero.zheng@huawei.com

Guangying Zheng
Huawei Technologies
China

Email: zhengguangying@huawei.com

Greg Mirsky
ZTE Corp.
USA

Email: gregimirsky@gmail.com

Reshad Rahman
Cisco Systems
Canada

Email: rrahman@cisco.com

Faisal Iqbal
Cisco Systems

Email: faiqbal@cisco.com