

Routing Area Working Group
Internet-Draft
Intended status: Informational
Expires: January 3, 2019

J. Dong
S. Bryant
Huawei
Z. Li
China Mobile
T. Miyasaka
KDDI Corporation
July 2, 2018

Enhanced Virtual Private Networks (VPN+)
draft-dong-teas-enhanced-vpn-00

Abstract

This draft describes a number of enhancements that need to be made to virtual private networks (VPNs) to support the needs of new applications, particularly applications that are associated with 5G services. A network enhanced with these properties may form the underpin of network slicing, but will also be of use in its own right.

Editor's Note: This is draft-bryant-rtgwg-enhanced-vpn moved to the TEAS WG.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 3, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
2. Requirements Language	4
3. Overview of the Requirements	4
3.1. Isolation between Virtual Networks	4
3.2. Diverse Performance Guarantees	6
3.3. A Pragmatic Approach to Isolation	7
3.4. Integration	8
3.5. Dynamic Configuration	9
3.6. Customized Control Plane	9
4. Applicability	10
5. Architecture and Components of Enhanced VPN	10
5.1. Communications Layering	10
5.2. Multi-Point to Multi-point	13
5.3. Candidate Underlay Technologies	13
5.3.1. FlexE	14
5.3.2. Dedicated Queues	14
5.3.3. Time Sensitive Networking	15
5.3.4. Deterministic Networking	15
5.3.5. MPLS Traffic Engineering (MPLS-TE)	15
5.3.6. Segment Routing	16
5.4. Control Plane Considerations	19
5.5. Application Specific Network Types	19
5.6. Integration with Service Functions	20
6. Scalability Considerations	20
6.1. Maximum Stack Depth	21
6.2. RSVP Scalability	21
7. OAM and Instrumentation	21
8. Enhanced Resiliency	22
9. Security Considerations	23
10. IANA Considerations	23
11. References	23
11.1. Normative References	23
11.2. Informative References	23
Authors' Addresses	25

1. Introduction

Virtual networks, often referred to as virtual private networks (VPNs) have served the industry well as a means of providing different groups of users with logically isolated access to a common network. The common or base network that is used to provide the VPNs is often referred to as the underlay, and the VPN is often called an overlay.

Driven largely by needs surfacing from 5G, the concept of network slicing has gained traction. There is a need to create a VPN with enhanced characteristics. Specifically there is a need for a transport network supporting a set of virtual networks each of which provides the client with dedicated (private) networking, computing and storage resources drawn from a shared pool. The tenant of such a network can require a degree of isolation and performance that previously could only be satisfied by dedicated networks. Additionally the tenant may ask for some level of control of their virtual network e.g. to customize the service paths in the network slice.

These properties cannot be met with pure overlay networks, as they require tighter coordination and integration between the underlay and the overlay network. This document introduces a new network service called enhanced VPN (VPN+). VPN+ refers to a virtual network which has dedicated network resources allocated from the underlay network. Unlike traditional VPN, an enhanced VPN can achieve greater isolation and guaranteed performance.

These new network layer properties, which have general applicability, may also be of interest as part of a network slicing solution.

This document specifies a framework for using the existing, modified and potential new networking technologies as components to provide an enhanced VPN (VPN+) service. Specifically we are concerned with:

- o The design of the enhanced VPN data-plane
- o The necessary protocols in both underlay and the overlay of enhanced VPN, and
- o The mechanisms to achieve integration between overlay and underlay
- o The necessary method of monitoring an enhanced VPN
- o The methods of instrumenting an enhanced VPN to ensure that the required tenant Service Level Agreement (SLA) is maintained

The required layer structure necessary to achieve this is shown in Section 5.1.

One use for enhanced VPNs is to create network slices with different isolation requirements. Such slices may be used to provide different tenants of vertical industrial markets with their own virtual network with the explicit characteristics required. These slices may be "hard" slices providing a high degree of confidence that the VPN+ characteristics will be maintained over the slice life cycle, or they may be "soft" slices in which case some degree of interaction may be experienced.

2. Requirements Language

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

3. Overview of the Requirements

In this section we provide an overview of the requirements of an enhanced VPN.

3.1. Isolation between Virtual Networks

The requirement is to provide both hard and soft isolation between the tenants/applications using one enhanced VPN and the tenants/applications using another enhanced VPN. Hard isolation is needed so that applications with exacting requirements can function correctly despite a flash demand being created on another VPN competing for the underlying resources. An example might be a network supporting both emergency services and public broadband multi-media services.

During a major incident the VPNs supporting these services would both be expected to experience high data volumes, and it is important that both make progress in the transmission of their data. In these circumstances the VPNs would require an appropriate degree of isolation to be able to continue to operate acceptably.

We introduce the terms hard (static) and soft (dynamic) isolation to cover cases such as the above. A VPN has soft isolation if the traffic of one VPN cannot be inspected by the traffic of another. Both IP and MPLS VPNs are examples of soft isolated VPNs because the network delivers the traffic only to the required VPN endpoints. However the traffic from one or more VPNs and regular network traffic may congest the network resulting in delays for other VPNs operating normally. The ability for a VPN to be sheltered from this effect is

called hard isolation, and this property is required by some critical applications. Although these isolation requirements are triggered by the needs of 5G networks, they have general utility. In the remainder of this section we explore how isolation may be achieved in packet networks.

It is of course possible to achieve high degrees of isolation in the optical layer. However this is done at the cost of allocating resources on a long term basis and end-to-end basis. Such an arrangement means that the full cost of the resources must be borne by the service that is allocated the resources. On the other hand, isolation at the packet layer allows the resources to be shared amongst many services and only dedicated to a service on a temporary basis. This allows greater statistical multiplexing of network resources and amortizes the cost over many services, leading to better economy. However, the degree of isolation required by network slicing cannot easily be met with MPLS-TE packet LSPs as they guarantee long-term bandwidth, but not latency.

Thus some trade-off between the two approaches needs to be considered to provide the required isolation between virtual networks while still allows reasonable sharing inside each VPN.

The work of the IEEE project on Time Sensitive Networking is introducing the concept of packet scheduling where a high priority packet stream may be given a scheduled time slot thereby guaranteeing that it experiences no queuing delay and hence a reduced latency. However where no scheduled packet arrives its reserved time-slot is handed over to best effort traffic, thereby improving the economics of the network. Such a scheduling mechanism may be usable directly, or with extension to achieve isolation between multiple VPNs.

One of the key areas in which isolation needs to be provided is at the interfaces. If nothing is done the system falls back to the router queuing system in which the ingress places it on a selected output queue. Modern routers have quite sophisticated output queuing systems, traditionally these have not provided the type of scheduling system needed to support the levels of isolation required by the applications that are the target of VPN+ networks. However some of the more modern approaches to queuing allow the construction of virtual channelized sub-interfaces (VCSI). With VCSIs there is only one physical interface, but the queuing system is used to provide virtual interfaces with dedicated resources. Sophisticated queuing systems of this type may be used to provide end-to-end virtual isolation between tenant's traffic in an otherwise homogeneous network.

[FLEXE] provides the ability to multiplex multiple channels over an Ethernet link in a way that provides hard isolation. However it is only a link technology. When packets are received by the downstream node they need to be processed in a way that preserves that isolation. This in turn requires a queuing and forwarding implementation that preserves the isolation end-to-end.

3.2. Diverse Performance Guarantees

There are several aspects to guaranteed performance: guaranteed maximum packet loss, guaranteed maximum delay and guaranteed delay variation.

Guaranteed maximum packet loss is a common parameter, and is usually addressed by setting the packet priorities, queue size and discard policy. However this becomes more difficult when the requirement is combine with the latency requirement. The limiting case is zero congestion loss, and that is the goal of the Deterministic Networking work that the IETF and IEEE are pursuing. In modern optical networks loss due to transmission errors is already asymptotic to zero due, but there is always the possibility of failure of the interface and the fiber itself. This can only be addressed by some form of packet duplication and transmission over diverse paths.

Guaranteed maximum latency is required in a number of applications particularly real-time control applications and some types of virtual reality applications. The work of the IETF Deterministic Networking (DetNet) Working Group is relevant, however the scope needs to be extended to methods of enhancing the underlay to better support the delay guarantee, and to integrate these enhancements with the overall service provision.

Guaranteed maximum delay variation is a service that may also be needed. Time transfer is one example of a service that needs this, although the fungible nature of time means that it might be delivered by the underlay as a shared service and not provided through different virtual networks. Alternatively a dedicated virtual network may be used to provide this as a shared service. The need for guaranteed maximum delay variation as a general requirement is for further study.

This leads to the concept that there is a spectrum of grades of service guarantee that need to be considered when deploying an enhanced VPN. As a guide to understanding the design requirements we can consider four types:

- o Guaranteed latency

- o Enhanced delivery
- o Assured bandwidth
- o Best effort

Best effort is the service that current VPNs provide. Providing assured bandwidth to VPNs, for example by using an RSVP-TE is not widely deployed at least partially due to scalability concerns. Guaranteed latency and enhanced delivery are not yet integrated with VPNs. It is these later two design requirements that enhanced VPNs provide.

In Section 3.1 we considered the work of the IEEE Time Sensitive Networking (TSN) project and the work of the IETF DetNet Working group in the context of isolation. However this work is of greater relevance in assuring end-to-end packet latency. It is also of importance in considering enhanced delivery.

A service that is guaranteed latency has a latency upper bound provided by the network. It is important to note that assuring the upper bound is more important than achieving the minimum latency.

A service that is offered enhanced delivery is one in which the network (at layer 3) attempts to deliver the packet through multiple paths in the hope of avoiding transient congestion [I-D.ietf-detnet-dp-sol].

A useful mechanism to provide these guarantees is to use Flex Ethernet [FLEXE] as the underlay. This is a method of bonding Ethernets together and of providing time-slot based channelization over an Ethernet bearer. Such channels are fully isolated from other channels running over the same Ethernet bearer. As noted elsewhere this produces hard isolation but at the cost of making the reclamation of unused bandwidth harder.

These approaches can usefully be used in tandem. For example, It is possible to use FlexE to provide tenant isolation, and then to use the TSN/Detnet approach over FlexE to provide service performance guarantee inside the a slice/tenant VPN.

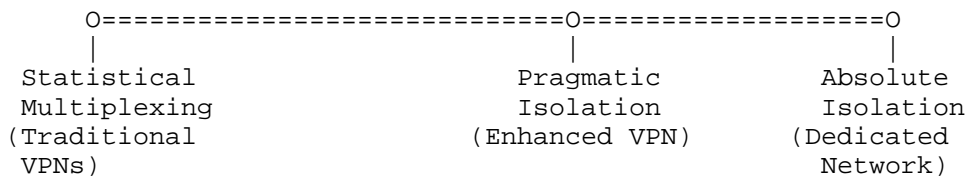
3.3. A Pragmatic Approach to Isolation

A key question to consider is whether it is possible to achieve hard isolation in packet networks. Packet networks were never designed to support hard isolation, just the opposite, they were designed to provide a high degree of statistical multiplexing and hence a significant economic advantage when compared to a dedicated, or a

Time Division Multiplexing (TDM) network. However the key thing to bear in mind is that the concept of hard isolation needs to be viewed from the perspective of the application, and there is no need to provide any harder isolation than is required by the application. From a historical perspective it is good to think about pseudowires [RFC3985] which emulate services that in many would have had hard isolation in their native form. However experience has shown that in most cases an approximation to this requirement is sufficient for most uses.

Thus, for example, using FlexE or channelized sub-interface, together with packet scheduling as interface slicing, and optionally, also together with the slicing of node resources (Network Processor Unit (NPU), etc.), it may be possible to provide a type of hard isolation that is adequate for many applications. Other applications may be satisfied with a classical VPN with or without reserved bandwidth, but yet others may require dedicated point to point fiber. The requirement is thus to qualify the needs of each application and provide an economic solution that satisfies those needs without over-engineering.

This spectrum of isolation is shown below:



At one end of the above figure, we have traditional statistical multiplexing technologies that support VPNs. This is a service type that has served the industry well and will continue to do so. At the opposite end of the spectrum we have the absolute isolation provided by traditional networks. The goal of enhanced VPN is pragmatic isolation. This is isolation that is better than is obtainable from pure statistical multiplexing, more cost effective and flexible than a dedicated network, but which is a practical solution that is good enough for the majority of applications.

3.4. Integration

A solution to the enhanced VPN problem will need to provide seamless integration of both overlay VPN and the underlay network resources. This needs to be done in a flexible and scalable way so that it can be widely deployed in operator networks. Given the targeting of both this technology and service function chaining at mobile networks and

in particular 5G the co-integration of service functions is a likely requirement.

3.5. Dynamic Configuration

It is necessary that new enhanced VPNs can be introduced to the network, modified, and removed from the network according to service demand. In doing so due regard must be given to the impact of other enhanced VPNs that are operational. An enhanced VPN that requires hard isolation must not be disrupted by the installation or modification of another enhanced VPN.

Whether modification of an enhanced VPN can be disruptive to that VPN, and in particular the traffic in flight is to be determined, but is likely to be a difficult problem to address.

The data-plane aspect of this are discussed further in Section 5.3.

The control-plane and management-plane aspects of this, particularly the garbage collection are likely to be challenging and are for further study.

As well as managing dynamic changes to the VPN in a seamless way, dynamic changes to the underlay and its transport network need to be managed in order to avoid disruption to sensitive services.

In addition to non-disruptively managing the network as a result of gross change such as the inclusion of a new VPN endpoint or a change to a link, consideration has to be given to the need to move VPN traffic as a result of traffic volume changes.

3.6. Customized Control Plane

In some cases it is desirable that an enhanced VPN has a custom control-plane, so that the tenant of the enhanced VPN can have some control to the resources and functions partitioned for this VPN. Each enhanced VPN may have its own dedicated controller, it may be provided with an interface to a control-plane that is shared with a set of other tenants, or it may be provided with an interface to the control-plane of the underlay provided by the underlay network operator.

Further detail on this requirement will be provided in a future version of the draft.

4. Applicability

The technologies described in this document is applicable to a number types of VPN technology such as:

- o Layer 2 point to point services such as pseudowires [RFC3985]
- o Layer 2 VPNs [RFC4664]
- o Ethernet VPNs [RFC7209]
- o Layer 3 VPNs [RFC4364], [RFC2764]

Where such VPN types need enhanced isolation and delivery characteristics the technology described here can be used to provide an underlay with the required enhanced performance.

5. Architecture and Components of Enhanced VPN

Normally a number of enhanced VPN services will be provided by a common network infrastructure. Each enhanced VPN consists of both the overlay and a specific set of dedicated network resources and functions allocated in the underlay to satisfy the needs of the VPN tenant. The integration between overlay and underlay ensures the isolation between different enhanced VPNs, and facilitates the guaranteed performance for different services.

An enhanced VPN needs to be designed with consideration given to:

- o Isolation of enhanced VPN data plane.
- o A scalable control plane to match the data plane isolation.
- o The amount of state in the packet vs the amount of state in the control plane.
- o Mechanism for diverse performance guarantee within an enhanced VPN
- o Support of the required integration between network functions and service functions.

5.1. Communications Layering

The communications layering model use to build an enhanced VPN is shown in Figure 1.

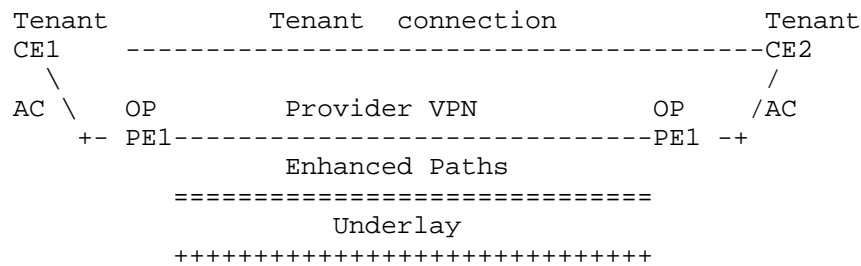


Figure 1: Communication Layering

The network operator is required to provide a tenant connection between the tenant's Customer Equipment (CE) (CE1 and CE2). These CEs attach to the Operator's Provider Edge Equipments (PE) (PE1 and PE2 respectively). The attachment circuits (AC) are outside the scope of this document other than to note that they obviously need to provide a connection of sufficient quality in terms of isolation, latency etc. so as to satisfy the needs of the user. The subtlety to be aware of is that the ACs are often provided by a network rather than a fixed point to point connection and thus the considerations in this document may apply to the network that provides the AC.

A provider VPN is constructed between PE1 and PE2 to carry tenant traffic. This is a normal VPN, and provides one stage of isolation between tenants.

An enhanced path is constructed to carry the provider VPN using dedicated resources drawn from the underlay.

This layered architecture is shown in more detail in Figure 2.

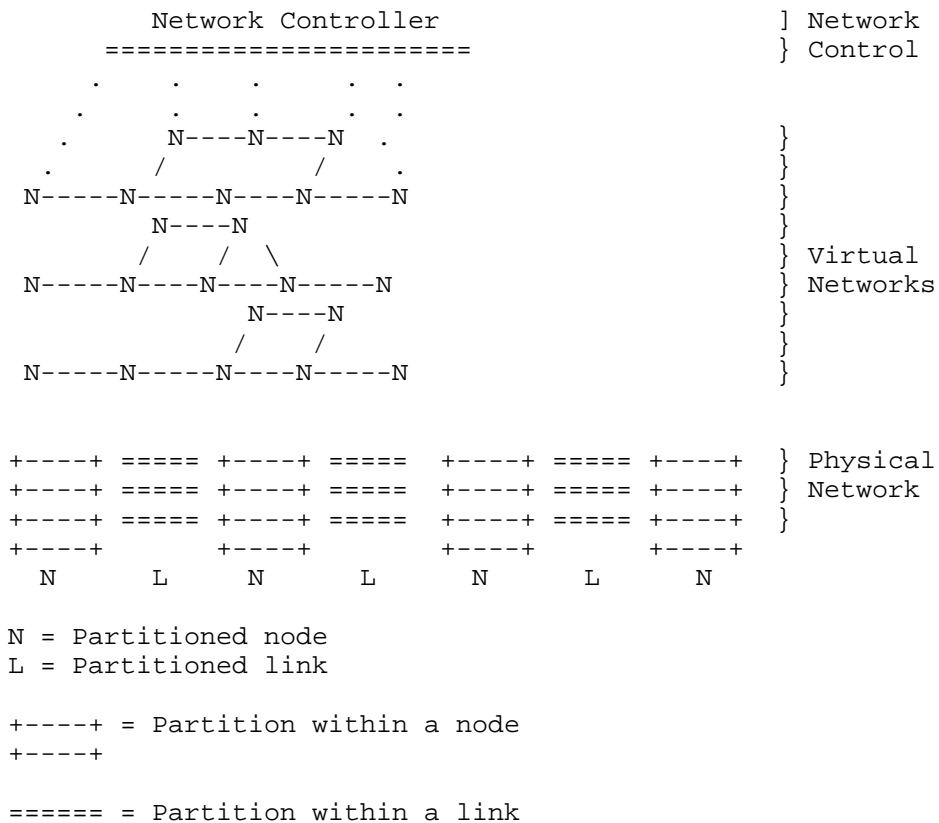


Figure 2: The Layers Architecture

Underpinning everything is the physical layer consisting of partitioned links and nodes which provide the underlying resources used to provision the logical networks. Various components and techniques as discussed in Section 5.3 are used to provide these resources, such as FlexE links, Time Sensitive Networking, Deterministic Networking etc. These partitions may be physical, or virtual so long as the SLA required by the higher layers is met.

These resources provision the virtual networks with dedicated resources that they need. To get the required functionality there needs to be integration between these overlays and the underlay providing the physical resources.

The network controller is used to create the virtual networks, to allocate the resources to each virtual network and to control and manage these networks.

The creation and allocation process needs to take a holistic view of the needs of all of its tenants, and to partition the resources accordingly. However within a virtual network these resources can if required be managed via a dynamic control plane. This provides the required scalability and isolation.

5.2. Multi-Point to Multi-point

At a VPN level connections are frequently multi-point-to-multi-point (MP2MP). As far as such services are concerned the underlay is an abstract MP2MP medium. However when service guarantees are provided, such as with an enhanced VPN, each point to point path through the underlay needs to be specifically engineered to meet the required performance guarantees.

5.3. Candidate Underlay Technologies

A VPN is a network created by applying a multiplexing technique to the underlying network (the underlay) in order to distinguish the traffic of one VPN from that of another. A VPN path that travels by other than the shortest path through the underlay normally requires state in the underlay to specify that path. State is normally applied to the underlay through the use of the RSVP Signaling protocol, or directly through the use of an SDN controller, although other techniques may emerge as this problem is studied. This state gets harder to manage as the number of VPN paths increases. Furthermore, as we increase the coupling between the underlay and the overlay to support the enhanced VPN service, this state will increase further.

In an enhanced VPN different subsets of the underlay resources are dedicated to different VPNs. Any enhanced VPN solution thus needs tighter coupling with underlay than is the case with classical VPNs. We cannot for example share the tunnel between enhanced VPNs which require hard isolation.

In the following sections we consider a number of candidate underlay solutions for proving the required VPN separation.

- o FlexE
- o Time Sensitive Networking
- o Deterministic Networking
- o Dedicated Queues

We then consider the problem of slice differentiation and resource representation. Candidate technologies are:

- o MPLS
- o MPLS-SR
- o Segment Routing over IPv6 (SRv6)

5.3.1. FlexE

FlexE [FLEXE] is a method of creating a point-to-point Ethernet with a specific fixed bandwidth. FlexE supports the bonding of multiple links, which supports creating larger links out of multiple slower links in a more efficient way than traditional link aggregation. FlexE also supports the sub-rating of links, which allows an operator to only use a portion of a link. FlexE also supports the channelization of links, which allows one link to carry several lower-speed or sub-rated links from different sources.

If different FlexE channels are used for different services, then no sharing is possible between the services. This in turn means that it is not possible to dynamically re-distribute unused bandwidth to lower priority services increasing the cost of operation of the network. FlexE can on the other hand be used to provide hard isolation between different tenants by providing hard isolation on an interface. The tenant can then use other methods to manage the relative priority of their own traffic.

Methods of dynamically re-sizing FlexE channels and the implication for enhanced VPN are under study.

5.3.2. Dedicated Queues

In an enhanced VPN providing multiple isolated virtual networks the conventional Diff-Serv based queuing system is insufficient for our purposes due to the limited number of queues which cannot differentiate between traffic of different VPNs and the range of service classes that each need to provide their tenants. This problem is particularly acute with an MPLS underlay due to the small number of traffic class services available. In order to address this problem and thus reduce the interference between VPNs, it is likely to be necessary to steer traffic of VPNs to dedicated input and output queues.

5.3.3. Time Sensitive Networking

Time Sensitive Networking (TSN) is an IEEE project that is designing a method of carrying time sensitive information over Ethernet. As Ethernet this can obviously be tunneled over a Layer 3 network in a pseudowire. However the TSN payload would be opaque to the underlay and thus not treated specifically as time sensitive data. The preferred method of carrying TSN over a layer 3 network is through the use of deterministic networking as explained in the following section of this document.

The mechanisms defined in TSN can be used to meet the requirements of time sensitive services of an enhanced VPN.

5.3.4. Deterministic Networking

Deterministic Networking (DetNet) [I-D.ietf-detnet-architecture] is a technique being developed in the IETF to enhance the ability of layer 3 networks to deliver packets more reliably and with greater control over the delay. The design cannot use classical re-transmission techniques such as TCP since can add delay that is above the maximum tolerated by the applications. Even the delay improvements that are achieved with SCTP-PR are outside the bounds set by application demands. The approach is to pre-emptively send copies of the packet over various paths in the expectation that this minimizes the chance of all packets being lost, but to trim duplicate packets to prevent excessive flooding of the network and to prevent multiple packets being delivered to the destination. It also seeks to set an upper bound on latency. Note that it is not the goal to minimize latency, and the optimum upper bound paths may not be the minimum latency paths.

DetNet is based on flows. It currently makes no comment on the underlay, and so at this stage must be assumed to use the base topology. To be of use in this application DetNet there needs to be a description of how to deal with the concept of flows within an enhanced VPN.

How we use DetNet in a multi-tenant (VPN) network, and how to improve the scalability of DetNet in a multi-tenant (VPN) network is for further study.

5.3.5. MPLS Traffic Engineering (MPLS-TE)

Normal MPLS runs on the base topology and has the concepts of reserving end to end bandwidth for an LSP, and of creating VPNs. VPN traffic can be run over dedicated RSVP-TE tunnels to provide reserved

bandwidth for a specific VPN connection. This is rarely deployed in practice due to scaling and management overhead concerns.

5.3.6. Segment Routing

Segment Routing [I-D.ietf-spring-segment-routing] is a method that prepends instructions to packets at entry and sometimes at various points as it passes through the network. These instructions allow packets to be routed on paths other than the shortest path for various traffic engineering reasons. These paths can be strict or loose paths, depending on the compactness required of the instruction list and the degree of autonomy granted to the network (for example to support ECMP).

With SR, a path needs to be dynamically created through a set of segments by simply specifying the Segment Identifiers (SIDs), i.e. instructions rooted at a particular point in the network. Thus if a path is to be provisioned from some ingress point A to some egress point B in the underlay, A is provided with the A..B SID list and instructions on how to identify the packets to which the SID list is to be prepended.

By encoding the state in the packet, as is done in Segment Routing, per-path state is transitioned out of the network.

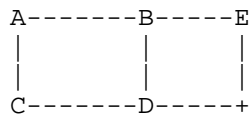


Figure 3: An SR Network Fragment

Consider the network fragment shown in Figure 3. To send a packet from A to E via B, D & E: Node A prepends the ordered list of SIDs (B, D, E) to the packet and pushes the packet to B. SID list {B, D, E} can be used as a VPN path. Thus, to create a VPN, a set of SID Lists is created and provided to each ingress node of the VPN together with packet selection criteria. In this way it is possible to create a VPN with no state in the core. However this is at the expense of creating a larger packet with possible MTU and hardware restriction limits that need to be overcome.

Note in the above if A and E support multiple VPN an additional VPN identifier will need to be added to the packet, but this is omitted from this text for simplicity.

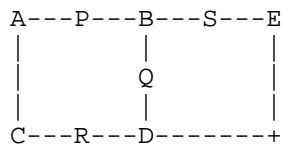


Figure 4: Another SR Network Fragment

Consider a further network fragment shown in Figure 4, and further consider VPN A+D+E.

A has lists: {P, B, Q, D}, {P, B, S, E}
 D has lists: {Q, B, P, A}, {E}
 E has lists: {S, B, P, A}, {D}

To create a new VPN C+D+B the following list are introduced:

C lists: {R, D}, {A, P, B}
 D lists: {R, C}, {Q, B}
 B lists: {Q, D}, {P, A, C}

Thus VPN C+D+B was created without touching the settings of the core routers, indeed it is possible to add endpoints to the VPNs, and move the paths around simply by providing new lists to the affected endpoints.

There are a number of limitations in SR as it is currently defined that limit its applicability to enhanced VPNs:

- o Segments are shared between different VPNs,
- o There is no reservation of bandwidth,
- o There is limited differentiation in the data plane.

Thus some extensions to SR are needed to provide isolation between different enhanced VPNs. This can be achieved by including a finer granularity of state in the core in anticipation of its future use by authorized services. We therefore need to evaluate the balance between this additional state and the performance delivered by the network.

Both MPLS Segment Routing and SRv6 Segment Routing are candidate technologies for enhanced VPN.

With current segment routing, the instructions are used to specify the nodes and links to be traversed. However, in order to achieve the required isolation between different services, new instructions

can be created which can be prepended to a packet to steer it through specific dedicated network resources and functions, e.g. links, queues, processors, services etc.

Clearly we can use traditional constructs to create a VPN, but there are advantages to the use of other constructs such as Segment Routing (SR) in the creation of virtual networks with enhanced properties.

Traditionally a traffic engineered path operates with a granularity of a link with hints about priority provided through the use of the traffic class field in the header. However to achieve the latency and isolation characteristics that are sought by the enhanced VPN users, steering packets through specific queues and resources will likely be required. The extent to which these needs can be satisfied through existing QoS mechanisms is to be determined. What is clear is that a fine control of which services wait for which, with a fine granularity of queue management policy is needed. Note that the concept of a queue is a useful abstraction for many types of underlay mechanism that may be used to provide enhanced isolation and latency support. From the perspective of the control plane and from the perspective of the segment routing the method of steering a packet to a queue that provides the required properties is a universal construct. How the queue satisfies the requirement is implementation specific and is transparent to the control plane and data plane mechanisms used. Thus for example a FlexE channel, or time sensitive networking packet scheduling slot are abstracted to the same concept and bound to the data plane in a common manner.

We can introduce the specification of finer, deterministic, granularity to path selection through extensions to traditional path construction techniques such as RSVP-TE and MPLS-TP.

We can also introduce it by specifying the queues through an SR instruction list. Thus new SR instructions may be created to specify not only which resources are traversed, but in some cases how they are traversed. For example, it may be possible to specify not only the queue to be used but the policy to be applied when enqueueing and dequeuing.

This concept can be further generalized, since as well as queuing to the output port of a router, it is possible to queue to any resource, for example:

- o A network processor unit (NPU)
- o A Central Processing Unit (CPU) Core
- o A Look-up engine such as TCAMs

5.4. Control Plane Considerations

It is expected that enhanced VPN would be based on a hybrid control mechanism, which takes advantage of the logically centralized controller for on-demand provisioning and global optimization, whilst still relies on distributed control plane to provide scalability, high reliability, fast reaction, automatic failure recovery etc. Extension and optimization to the distributed control plane is needed to support the enhanced properties of VPN+.

Where SR is used as a the data-plane construct it needs to be noted that it does not have the capability of reserving resources along the path nor do its currently specified distributed control plane (the link state routing protocols). An SDN controller can clearly do this, from the controllers point of view, and no resource reservation is done on the device. Thus if a distributed control plane is needed either in place of an SDN controller or as an assistant to it, the design of the control system needs to ensure that resources are uniquely allocated to the correct service, and no allocated to multiple services causing unintended resource conflict. This needs further study.

On the other hand an advantage of using an SR approach is that it provides a way of efficiently binding the network underlay and the enhanced VPN overlay. With a technology such as RSVP-TE LSPs, each virtual path in the VPN is bound to the underlay with a dedicated TE-LSP.

RSVP-TE could be enhanced to bind the VPN to specific resources within the underlay, but as noted elsewhere in this document there are concerns as to the scalability of this approach. With an SR-based approach to resource reservation (per-slice reservation), it is straightforward to create dedicated SR network slices, and the VPN can be bound to a particular SR network slice.

5.5. Application Specific Network Types

Although a lot of the traffic that will be carried over the enhanced VPN will likely be IPv4 or IPv6, the design has to be capable of carrying other traffic types. In particular the design SHOULD be capable of carrying Ethernet traffic. This is easily accomplished through the various pseudowire (PW) techniques [RFC3985]. Where the underlay is MPLS Ethernet can be carried over the enhanced VPN encapsulated according to the method specified in [RFC4448]. Where the underlay is IP Layer Two Tunneling Protocol - Version 3 (L2TPv3) [RFC3931] can be used with Ethernet traffic carried according to [RFC4719]. Encapsulations have been defined for most of the common layer two type for both PW over MPLS and for L2TPv3.

5.6. Integration with Service Functions

There is a significant overlap between the problem of routing a packet through a set of network resources and the problem of routing a packet through a set of compute resources. Service Function Chain technology is designed to forward a packet through a set of compute resources.

A future version of this document will discuss this further.

6. Scalability Considerations

For a packet to transit a network, other than on a best effort, shortest path basis, it is necessary to introduce additional state, either in the packet, or in the network or some combination of both.

There are at least three ways of doing this:

- o Introduce the complete state into the packet. That is how SR does this, and this allows the controller to specify the precise series of forwarding and processing instructions that will happen to the packet as it transits the network. The cost of this is an increase in the packet header size. The cost is also that systems will have capabilities enabled in case they are called upon by a service. This is a type of latent state, and increases as we more precisely specify the path and resources that need to be exclusively available to a VPN.
- o Introduce the state to the network. This is normally done by creating a path using RSVP-TE, which can be extended to introduce any element that needs to be specified along the path, for example explicitly specifying queuing policy. It is of course possible to use other methods to introduce path state, such as via a Software Defined Network (SDN) controller, or possibly by modifying a routing protocol. With this approach there is state per path per path characteristic that needs to be maintained over its life-cycle. This is more state than is needed using SR, but the packet are shorter.
- o Provide a hybrid approach based on using binding SIDs to create path fragments, and bind them together with SR.

Dynamic creation of a VPN path using SR requires less state maintenance in the network core at the expense of larger VPN headers on the packet. The scaling properties will reduce roughly from a function of $(N/2)^2$ to a function of N , where N is the VPN path length in intervention points (hops plus network functions). Reducing the state in the network is important to VPN+, as VPN+

requires the overlay to be more closely integrated with the underlay than with traditional VPNs. This tighter coupling would normally mean that significant state needed to be created and maintained in the core. However, a segment routed approach allows much of this state to be spread amongst the network ingress nodes, and transiently carried in the packets as SIDs.

These approaches are for further study.

6.1. Maximum Stack Depth

One of the challenges with SR is the stack depth that nodes are able to impose on packets. This leads to a difficult balance between adding state to the network and minimizing stack depth, or minimizing state and increasing the stack depth.

6.2. RSVP Scalability

The traditional method of creating a resource allocated path through an MPLS network is to use the RSVP protocol. However there have been concerns that this requires significant continuous state maintenance in the network. There are ongoing works to improve the scalability of RSVP-TE LSPs in the control plane [I-D.ietf-teas-rsvp-te-scaling-rec]. This will be considered further in a future version of this document.

There is also concern at the scalability of the forwarder footprint of RSVP as the number of paths through an LSR grows [I-D.sitaraman-mpls-rsvp-shared-labels] proposes to address this by employing SR within a tunnel established by RSVP-TE. This work will be considered in a future version of this document.

7. OAM and Instrumentation

A study of OAM in SR networks has been documented in [I-D.ietf-spring-oam-usecase].

The enhanced VPN OAM design needs to consider the following requirements:

- o Instrumentation of the underlay so that the network operator can be sure that the resources committed to a tenant are operating correctly and delivering the required performance.
- o Instrumentation of the overlay by the tenant. This is likely to be transparent to the network operator and to use existing methods. Particular consideration needs to be given to the need

to verify the isolation and the various committed performance characteristics.

- o Instrumentation of the overlay by the network provider to proactively demonstrate that the committed performance is being delivered. This needs to be done in a non-intrusive manner, particularly when the tenant is deploying a performance sensitive application
- o Verification of the conformity of the path to the service requirement. This may need to be done as part of a commissioning test.

These issues will be discussed in a future version of this document.

8. Enhanced Resiliency

Each enhanced VPN, of necessity, has a life-cycle, and needs modification during deployment as the needs of its user change. Additionally as the network as a whole evolves there will need to be garbage collection performed to consolidate resources into usable quanta.

Systems in which the path is imposed such as SR, or some form of explicit routing tend to do well in these applications because it is possible to perform an atomic transition from one path to another. However implementations and the monitoring protocols need to make sure that the new path is up before traffic is transitioned to it.

There are however two manifestations of the latency problem that are for further study in any of these approaches:

- o The problem of packets overtaking one and other if a path latency reduces during a transition.
- o The problem of the latency transient in either direction as a path migrates.

There is also the matter of what happens during failure in the underlay infrastructure. Fast reroute is one approach, but that still produces a transient loss with a normal goal of rectifying this within 50ms. An alternative is some form of N+1 delivery such as has been used for many years to support protection from service disruption. This may be taken to a different level using the techniques proposed by the IETF deterministic network work with multiple in-network replication and the culling of later packets.

In addition to the approach used to protect high priority packets, consideration has to be given to the impact of best effort traffic on the high priority packets during a transient. Specifically if a conventional re-convergence process is used there will inevitably be micro-loops and whilst some form of explicit routing will protect the high priority traffic, lower priority traffic on best effort shortest paths will micro-loop without the use of a loop prevention technology. To provide the highest quality of service to high priority traffic, either this traffic must be shielded from the micro-loops, or micro-loops must be prevented.

9. Security Considerations

All types of virtual network require special consideration to be given to the isolation between the tenants. However in an enhanced virtual network service hard isolation needs to be considered. If a service requires a specific latency then it can be damaged by simply delaying the packet through the activities of another tenant. In a network with virtual functions, depriving a function used by another tenant of compute resources can be just as damaging as delaying transmission of a packet in the network.

10. IANA Considerations

There are no requested IANA actions.

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>>.

11.2. Informative References

[FLEXE] "Flex Ethernet Implementation Agreement", March 2016, <<http://www.oiforum.com/wp-content/uploads/OIF-FLEXE-01.0.pdf>>.

[I-D.ietf-detnet-architecture] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-05 (work in progress), May 2018.

- [I-D.ietf-detnet-dp-sol]
Korhonen, J., Andersson, L., Jiang, Y., Finn, N., Varga, B., Farkas, J., Bernardos, C., Mizrahi, T., and L. Berger, "DetNet Data Plane Encapsulation", draft-ietf-detnet-dp-sol-04 (work in progress), March 2018.
- [I-D.ietf-spring-oam-usecase]
Geib, R., Filsfils, C., Pignataro, C., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", draft-ietf-spring-oam-usecase-10 (work in progress), December 2017.
- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.
- [I-D.ietf-teas-rsvp-te-scaling-rec]
Beeram, V., Minei, I., Shakir, R., Pacella, D., and T. Saad, "Techniques to Improve the Scalability of RSVP Traffic Engineering Deployments", draft-ietf-teas-rsvp-te-scaling-rec-09 (work in progress), February 2018.
- [I-D.sitaraman-mpls-rsvp-shared-labels]
Sitaraman, H., Beeram, V., Parikh, T., and T. Saad, "Signaling RSVP-TE tunnels on a shared MPLS forwarding plane", draft-sitaraman-mpls-rsvp-shared-labels-03 (work in progress), December 2017.
- [RFC2764] Gleeson, B., Lin, A., Heinanen, J., Armitage, G., and A. Malis, "A Framework for IP Based Virtual Private Networks", RFC 2764, DOI 10.17487/RFC2764, February 2000, <<https://www.rfc-editor.org/info/rfc2764>>.
- [RFC3931] Lau, J., Ed., Townsley, M., Ed., and I. Goyret, Ed., "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, DOI 10.17487/RFC3931, March 2005, <<https://www.rfc-editor.org/info/rfc3931>>.
- [RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <<https://www.rfc-editor.org/info/rfc3985>>.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <<https://www.rfc-editor.org/info/rfc4364>>.

- [RFC4448] Martini, L., Ed., Rosen, E., El-Aawar, N., and G. Heron, "Encapsulation Methods for Transport of Ethernet over MPLS Networks", RFC 4448, DOI 10.17487/RFC4448, April 2006, <<https://www.rfc-editor.org/info/rfc4448>>.
- [RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, DOI 10.17487/RFC4664, September 2006, <<https://www.rfc-editor.org/info/rfc4664>>.
- [RFC4719] Aggarwal, R., Ed., Townsley, M., Ed., and M. Dos Santos, Ed., "Transport of Ethernet Frames over Layer 2 Tunneling Protocol Version 3 (L2TPv3)", RFC 4719, DOI 10.17487/RFC4719, November 2006, <<https://www.rfc-editor.org/info/rfc4719>>.
- [RFC7209] Sajassi, A., Aggarwal, R., Uttaro, J., Bitar, N., Henderickx, W., and A. Isaac, "Requirements for Ethernet VPN (EVPN)", RFC 7209, DOI 10.17487/RFC7209, May 2014, <<https://www.rfc-editor.org/info/rfc7209>>.

Authors' Addresses

Jie Dong
Huawei

Email: jie.dong@huawei.com

Stewart Bryant
Huawei

Email: stewart.bryant@gmail.com

Zhenqiang Li
China Mobile

Email: lizhenqiang@chinamobile.com

Takuya Miyasaka
KDDI Corporation

Email: ta-miyasaka@kddi.com

TEAS Working Group
Internet Draft
Intended Status: Standard Track
Expires: December 19, 2018

Y. Lee (Editor)
Dhruv Dhody
Huawei
D. Ceccarelli
Ericsson
Igor Bryskin
Huawei
Bin Yeong Yoon
ETRI
Qin Wu
Huawei
Peter Park
KT

June 19, 2018

A Yang Data Model for ACTN VN Operation

draft-ietf-teas-actn-vn-yang-01

Abstract

This document provides a YANG data model for the Abstraction and Control of Traffic Engineered (TE) networks (ACTN) Virtual Network Service (VNS) operation.

Status of this Memo

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

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

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

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

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

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

Copyright Notice

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

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

Table of Contents

1. Introduction.....	3
1.1. Terminology.....	4
2. ACTN CMI context.....	4
2.1. Type 1 VN.....	4
2.2. Type 2 VN.....	5
3. High-Level Control Flows with Examples.....	7
3.1. Type 1 VN Illustration.....	7
3.2. Type 2 VN Illustration.....	8
4. Justification of the ACTN VN Model on the CMI.....	10
4.1. Customer view of VN.....	10
4.2. Innovative Services.....	11
4.2.1. VN Compute.....	11
4.2.2. Multi-sources and Multi-destinations.....	11
4.2.3. Others.....	12
4.3. Summary.....	12
5. ACTN VN YANG Model (Tree Structure).....	13
6. ACTN-VN YANG Code.....	15
7. JSON Example.....	27
7.1. ACTN VN JSON.....	28
7.2. TE-topology JSON.....	34
8. Security Considerations.....	50
9. IANA Considerations.....	50
10. Acknowledgments.....	50

11. References.....	51
11.1. Normative References.....	51
11.2. Informative References.....	51
12. Contributors.....	52
Authors' Addresses.....	52

1. Introduction

This document provides a YANG data model for the Abstraction and Control of Traffic Engineered (TE) networks (ACTN) Virtual Network Service (VNS) operation that is going to be implemented for the Customer Network Controller (CNC)- Multi-Domain Service Coordinator (MSDC) interface (CMI).

The YANG model on the CMI is also known as customer service model in [RFC8309]. The YANG model discussed in this document is used to operate customer-driven VNs during the VN instantiation, VN computation, and its life-cycle service management and operations.

The VN model defined in this document can also work together with other customer service models such as L3SM [RFC8299], L2SM [L2SM] and L1CSM [L1CSM] to provide a complete life-cycle service management and operations.

The YANG model discussed in this document basically provides the following:

- o Characteristics of Access Points (APs) that describe customer's end point characteristics;
- o Characteristics of Virtual Network Access Points (VNAP) that describe How an AP is partitioned for multiple VNs sharing the AP and its reference to a Link Termination Point (LTP) of the Provider Edge (PE) Node;
- o Characteristics of Virtual Networks (VNs) that describe the customer's VNs in terms of VN Members comprising a VN, multi-source and/or multi-destination characteristics of VN Member, the VN's reference to TE-topology's Abstract Node;

The actual VN instantiation and computation is performed with Connectivity Matrices sub-module of TE-Topology Model [TE-Topo] which provides TE network topology abstraction and management operation. Once TE-topology Model is used in triggering VN instantiation over the networks, TE-tunnel [TE-tunnel] Model will inevitably interact with TE-Topology model for setting up actual tunnels and LSPs under the tunnels.

The ACTN VN operational state is included in the same tree as the configuration consistent with Network Management Datastore Architecture (NMDA) [NMDA]. The origin of the data is indicated as per the origin metadata annotation.

1.1. Terminology

Refer to [ACTN-Frame], [RFC7926], and [RFC8309] for the key terms used in this document.

2. ACTN CMI context

The model presented in this document has the following ACTN context.

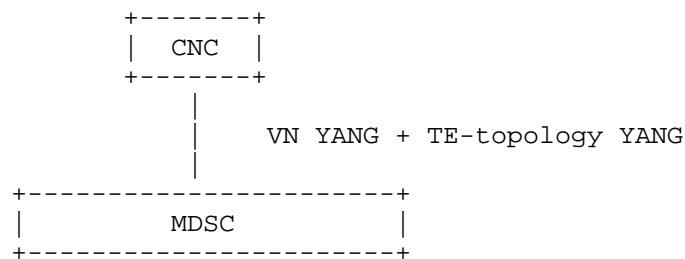


Figure 1. ACTN CMI

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

2.1. Type 1 VN

As defined in [ACTN-FW], a Virtual Network is a customer view of the TE network. To recapitulate VN types from [ACTN-FW], Type 1 VN is defined as follows:

The VN can be seen as a set of edge-to-edge abstract links (a Type 1 VN). Each abstract link is referred to as a VN member and is formed as an end-to-end tunnel across the underlying networks. Such tunnels may be constructed by recursive slicing or abstraction of paths in the underlying networks and can encompass edge points of the

customer's network, access links, intra-domain paths, and inter-domain links.

If we were to create a VN where we have four VN-members as follows:

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

Where L1, L2, L3, L4, L7 and L8 correspond to a Customer End-Point, respectively.

This VN can be modeled as one abstract node representation as follows in Figure 2:

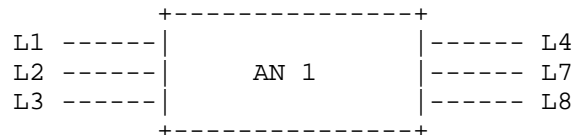


Figure 2. Abstract Node (One node topology)

Modeling a VN as one abstract node is the easiest way for customers to express their end-to-end connectivity; however, customers are not limited to express their VN only with one abstract node. In some cases, more than one abstract nodes can be employed to express their VN.

2.2. Type 2 VN

For some VN members of a VN, the customers are allowed to configure the actual path (i.e., detailed virtual nodes and virtual links) over the VN/abstract topology agreed mutually between CNC and MDSC prior to or a topology created by the MDSC as part of VN instantiation. Type 2 VN is always built on top of a Type 1 VN.

If a Type 2 VN is desired for some or all of VN members of a type 1 VN (see the example in Section 2.1), the TE-topology model can

provide the following abstract topology (that consists of virtual nodes and virtual links) which is built on top of the Type 1 VN.

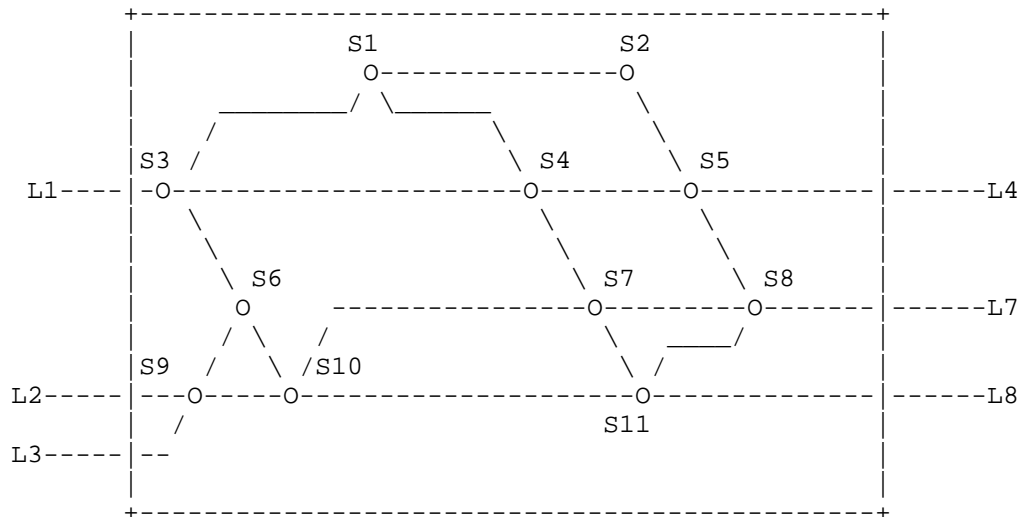


Figure 3. Type 2 topology

As you see from Figure 3, the Type 1 abstract node is depicted as a Type 1 abstract topology comprising of detailed virtual nodes and virtual links.

As an example, if VN-member 1 (L1-L4) is chosen to configure its own path over Type 2 topology, it can select, say, a path that consists of the ERO {S3,S4,S5} based on the topology and its service requirement. This capability is enacted via TE-topology configuration by the customer.

3. High-Level Control Flows with Examples

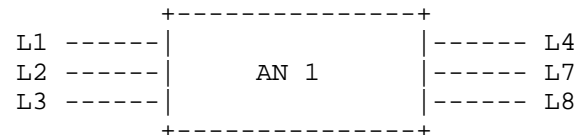
3.1. Type 1 VN Illustration

If we were to create a VN where we have four VN-members as follows:

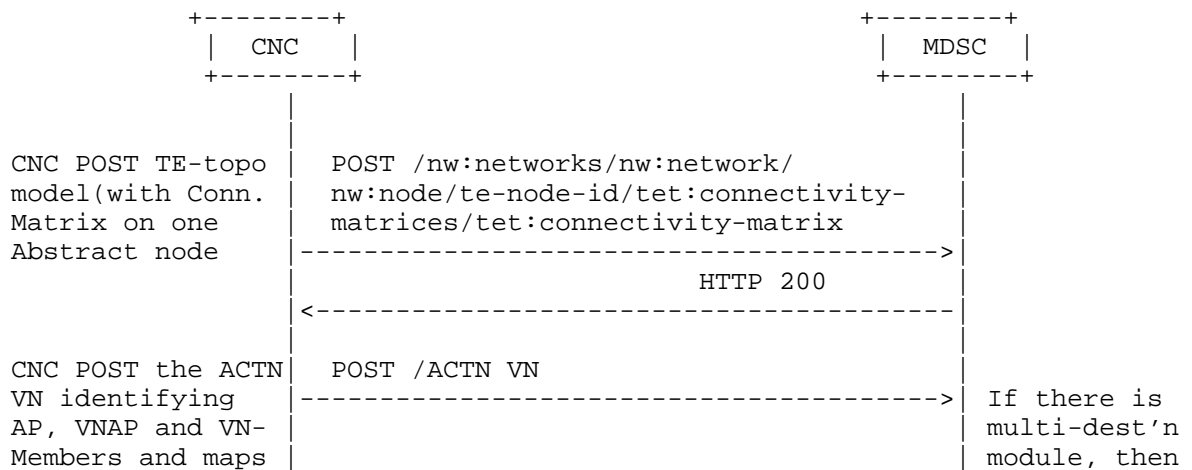
VN-Member 1	L1-L4
VN-Member 2	L1-L7
VN-Member 3	L2-L4
VN-Member 4	L3-L8

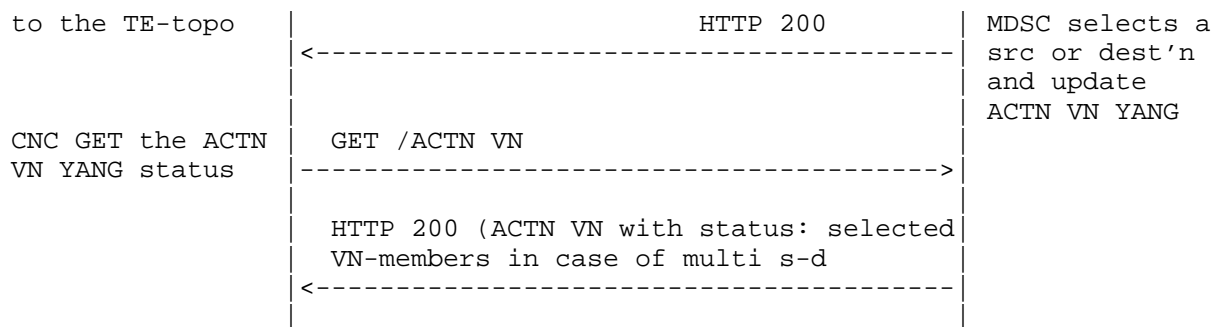
Where L1, L2, L3, L4, L7 and L8 correspond to Customer End-Point, respectively.

This VN can be modeled as one abstract node representation as follows:



If this VN is Type 1, the following diagram shows the message flow between CNC and MDSC to instantiate this VN using ACTN VN and TE-Topology Model.



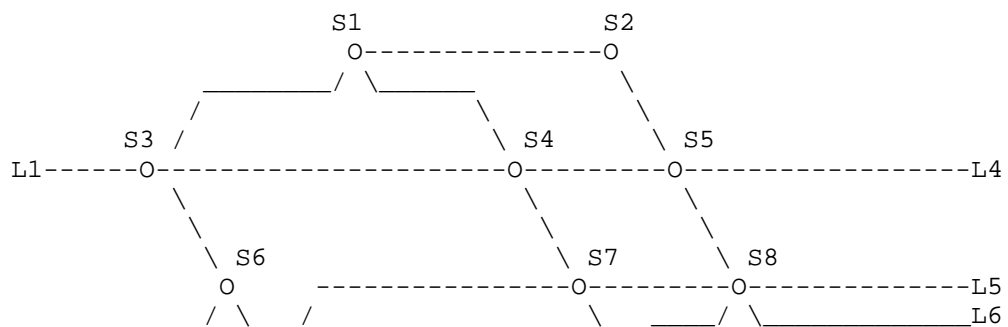


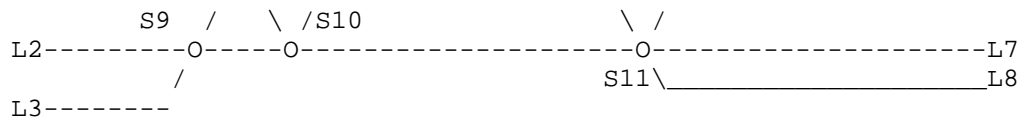
3.2. Type 2 VN Illustration

For some VN members, the customer may want to "configure" explicit routes over the path that connects its two end-points. Let us consider the following example.

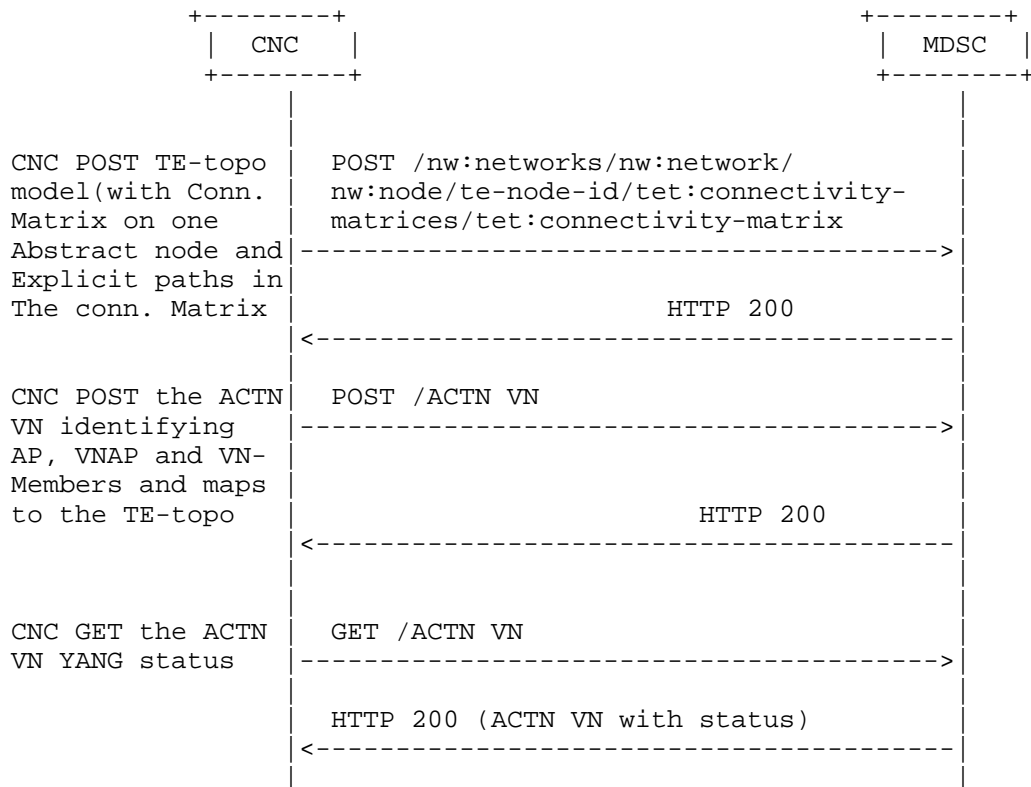
VN-Member 1	L1-L4
VN-Member 2	L1-L7 (via S4 and S7)
VN-Member 3	L2-L4
VN-Member 4	L3-L8 (via S10)

Where the following topology is the underlay for Abstraction Node 1 (AN1).

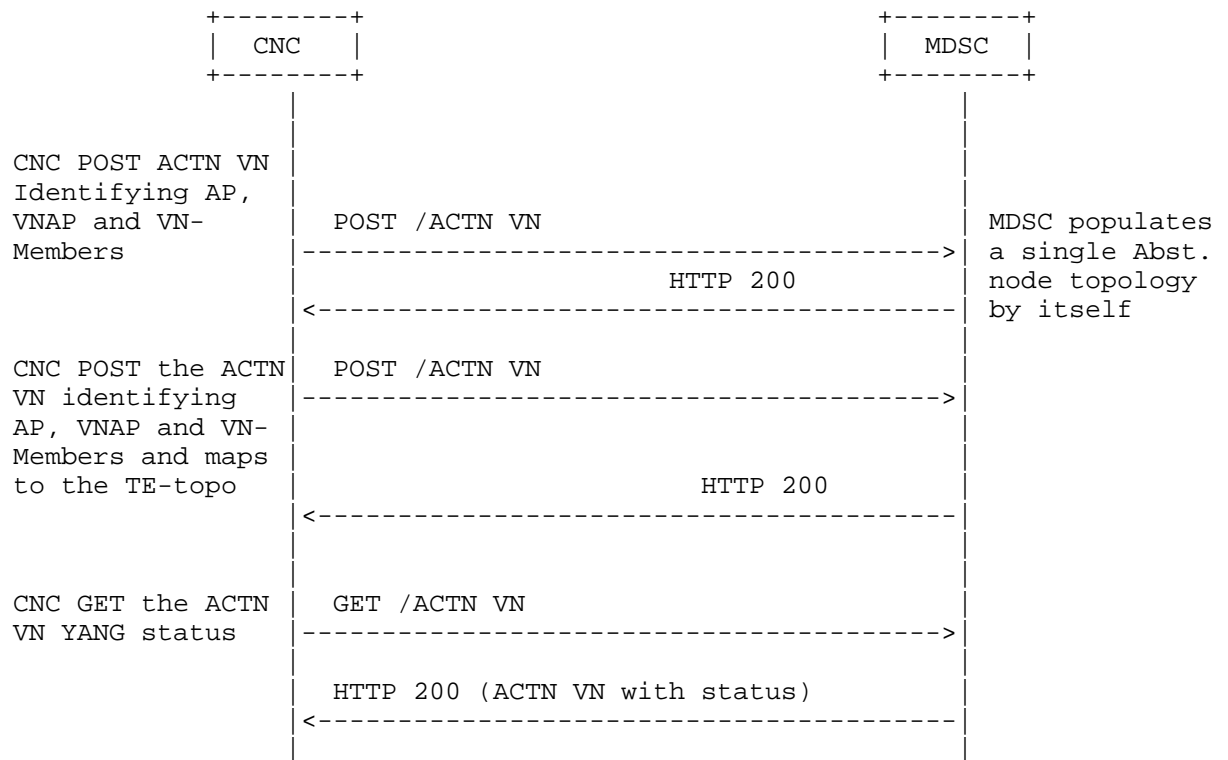




If CNC creates the single abstract topology, the following diagram shows the message flow between CNC and MDSC to instantiate this VN using ACTN VN and TE-Topology Model.



On the other hand, if MDSC create single node topology based ACTN VN YANG posted by the CNC, the following diagram shows the message flow between CNC and MDSC to instantiate this VN using ACTN VN and TE-Topology Model.



4. Justification of the ACTN VN Model on the CMI.

4.1. Customer view of VN

The VN-Yang model allows to define a customer view, and allows the customer to communicate using the VN constructs as described in the [ACTN-INFO]. It also allows to group the set of edge-to-edge links (i.e., VN members) under a common umbrella of VN. This allows the customer to instantiate and view the VN as one entity, making it easier for some customers to work on VN without worrying about the details of the provider based YANG models.

This is similar to the benefits of having a separate YANG model for the customer services as described in [SERVICE-YANG], which states that service models do not make any assumption of how a service is actually engineered and delivered for a customer.

4.2. Innovative Services

4.2.1. VN Compute

ACTN VN supports VN compute (pre-instantiation mode) to view the full VN as a single entity before instantiation. Achieving this via path computation or "compute only" tunnel setup does not provide the same functionality.

4.2.2. Multi-sources and Multi-destinations

In creating a virtual network, the list of sources or destinations or both may not be pre-determined by the customer. For instance, for a given source, there may be a list of multiple-destinations to which the optimal destination may be chosen depending on the network resource situations. Likewise, for a given destination, there may also be multiple-sources from which the optimal source may be chosen. In some cases, there may be a pool of multiple sources and destinations from which the optimal source-destination may be chosen. The following YANG module is shown for describing source container and destination container. The following YANG tree shows how to model multi-sources and multi-destinations.

```

+---rw actn
  . . .
  +---rw vn
    +---rw vn-list* [vn-id]
      +---rw vn-id          uint32
      +---rw vn-name?       string
      +---rw vn-topology-id? te-types:te-topology-id
      +---rw abstract-node?  -> /nw:networks/network/node/tet:te-node-id
      +---rw vn-member-list* [vn-member-id]
        | +---rw vn-member-id          uint32
        | +---rw src
        | | +---rw src?                -> /actn/ap/access-point-list/access-po
int-id   | |
ap-id    | | +---rw src-vn-ap-id?     -> /actn/ap/access-point-list/vn-ap/vn-
        | | +---rw multi-src?         boolean {multi-src-dest}?
        | | +---rw dest
        | | +---rw dest?              -> /actn/ap/access-point-list/access-p
oint-    | |
id       | | +---rw dest-vn-ap-id?    -> /actn/ap/access-point-list/vn-ap/vn-
-ap-id   | | +---rw multi-dest?       boolean {multi-src-dest}?

```

```

      |  +--rw connetivity-matrix-id?  -> /nw:networks/network/node/tet:
te/te-
node-attributes/connectivity-matrices/connectivity-matrix/id
      |  +--ro oper-status?             identityref
+--ro if-selected?      boolean {multi-src-dest}?
+--rw admin-status?     identityref
+--ro oper-status?      identityref

```

4.2.3. Others

The VN Yang model can be easily augmented to support the mapping of VN to the Services such as L3SM and L2SM as described in [TE-MAP].

The VN Yang model can be extended to support telemetry, performance monitoring and network autonomies as described in [ACTN-PM].

4.3. Summary

This section summarizes the innovative service features of the ACTN VN Yang.

- o Maintenance of AP and VNAP along with VN.
- o VN construct to group of edge-to-edge links
- o VN Compute (pre-instantiate)
- o Multi-Source / Multi-Destination
- o Ability to support various VN and VNS Types
 - * VN Type 1: Customer configures the VN as a set of VN Members.
No other details need to be set by customer, making for a simplified operations for the customer.
 - * VN Type 2: Along with VN Members, the customer could also provide an abstract topology, this topology is provided by the Abstract TE Topology Yang Model.

5. ACTN VN YANG Model (Tree Structure)

```

module: ietf-actn-vn
  +--rw actn
    +--rw ap
      +--rw access-point-list* [access-point-id]
        +--rw access-point-id      uint32
        +--rw access-point-name?   string
        +--rw max-bandwidth?       te-types:te-bandwidth
        +--rw avl-bandwidth?       te-types:te-bandwidth
        +--rw vn-ap* [vn-ap-id]
          +--rw vn-ap-id           uint32
          +--rw vn?                -> /actn/vn/vn-list/vn-id
          +--rw abstract-node?     ->
/nw:networks/network/node/tet:te-node-id
      +--rw ltp?                  te-types:te-tp-id
    +--rw vn
      +--rw vn-list* [vn-id]
        +--rw vn-id                uint32
        +--rw vn-name?             string
        +--rw vn-topology-id?      te-types:te-topology-id
        +--rw abstract-node?       ->
/nw:networks/network/node/tet:te-node-id
      +--rw vn-member-list* [vn-member-id]
        +--rw vn-member-id         uint32
        +--rw src
          +--rw src?               -> /actn/ap/access-point-
list/access-point-id
          +--rw src-vn-ap-id?      -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
          +--rw multi-src?         boolean {multi-src-dest}?
          +--rw dest
            +--rw dest?            -> /actn/ap/access-point-
list/access-point-id

```

```

        | | | +--rw dest-vn-ap-id?    -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
        | | | +--rw multi-dest?        boolean {multi-src-dest}?
        | | | +--rw connetivity-matrix-id?    ->
/nw:networks/network/node/tet:te/te-node-attributes/connectivity-
matrices/connectivity-matrix/id
        | | | +--ro oper-status?          identityref
        | | | +--ro if-selected?          boolean {multi-src-dest}?
        | | | +--rw admin-status?          identityref
        | | | +--ro oper-status?          identityref
        | | | +--rw vn-level-diversity?    vn-disjointness

rpcs:
  +---x vn-compute
    +---w input
      | +---w abstract-node?            ->
/nw:networks/network/node/tet:te-node-id
      | | +---w vn-member-list* [vn-member-id]
      | | | +---w vn-member-id          uint32
      | | | +---w src
      | | | | +---w src?                -> /actn/ap/access-point-
list/access-point-id
      | | | | +---w src-vn-ap-id?      -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
      | | | | +---w multi-src?          boolean {multi-src-dest}?
      | | | | +---w dest
      | | | | +---w dest?              -> /actn/ap/access-point-
list/access-point-id
      | | | | +---w dest-vn-ap-id?    -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
      | | | | +---w multi-dest?          boolean {multi-src-dest}?
      | | | | +---w connetivity-matrix-id?    ->
/nw:networks/network/node/tet:te/te-node-attributes/connectivity-
matrices/connectivity-matrix/id
      | | | +---w vn-level-diversity?    vn-disjointness
    +--ro output
      +---ro vn-member-list* [vn-member-id]
      +---ro vn-member-id          uint32
      +---ro src
      | +---ro src?                -> /actn/ap/access-point-
list/access-point-id

```

```

|   +--ro src-vn-ap-id?   -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
|   +--ro multi-src?      boolean {multi-src-dest}?
+--ro dest
|   +--ro dest?           -> /actn/ap/access-point-
list/access-point-id
|   +--ro dest-vn-ap-id?  -> /actn/ap/access-point-
list/vn-ap/vn-ap-id
|   +--ro multi-dest?     boolean {multi-src-dest}?
+--ro connectivity-matrix-id? ->
/nw:networks/network/node/tet:te/te-node-attributes/connectivity-
matrices/connectivity-matrix/id
+--ro if-selected?       boolean {multi-src-
dest}?
+--ro compute-status?    identityref

```

6. ACTN-VN YANG Code

The YANG code is as follows:

```

<CODE BEGINS> file "ietf-actn-vn@2018-02-27.yang"

module ietf-actn-vn {
  namespace "urn:ietf:params:xml:ns:yang:ietf-actn-vn";
  prefix "vn";

  /* Import network */
  import ietf-network {
    prefix "nw";
  }

  /* Import TE generic types */
  import ietf-te-types {
    prefix "te-types";
  }

  /* Import Abstract TE Topology */
  import ietf-te-topology {
    prefix "tet";
  }

  organization

```



```
"IETF Traffic Engineering Architecture and Signaling (TEAS)
Working Group";
contact
  "Editor: Young Lee <leeyoung@huawei.com>
    : Dhruv Dhody <dhruv.ietf@gmail.com>";
description
  "This module contains a YANG module for the ACTN VN. It
  describes a VN operation module that takes place in the
  context of the CNC-MDSC Interface (CMI) of the ACTN
  architecture where the CNC is the actor of a VN
  Instantiation/modification /deletion.";
revision 2018-02-27 {
  description
    "initial version.";
  reference
    "TBD";
}
/*
 * Features
 */
feature multi-src-dest {
  description
    "Support for selection of one src or destination
    among multiple.";
}

/*identity path-metric-delay {
  base te-types:path-metric-type;
  description
    "delay path metric";
}
identity path-metric-delay-variation {
  base te-types:path-metric-type;
  description
    "delay-variation path metric";
}
identity path-metric-loss {
  base te-types:path-metric-type;
  description
    "loss path metric";
}*/

identity vn-state-type {
  description
    "Base identity for VN state";
```

```
}
identity vn-state-up {
    base vn-state-type;
    description "VN state up";
}
identity vn-state-down {
    base vn-state-type;
    description "VN state down";
}
identity vn-admin-state-type {
    description
        "Base identity for VN admin states";
}
identity vn-admin-state-up {
    base vn-admin-state-type;
    description "VN administratively state up";
}
identity vn-admin-state-down {
    base vn-admin-state-type;
    description "VN administratively state down";
}
identity vn-compute-state-type {
    description
        "Base identity for compute states";
}
identity vn-compute-state-computing {
    base vn-compute-state-type;
    description
        "State path compute in progress";
}
identity vn-compute-state-computation-ok {
    base vn-compute-state-type;
    description
        "State path compute successful";
}
identity vn-compute-state-computatione-failed {
    base vn-compute-state-type;
    description
        "State path compute failed";
}
/*
 * Groupings
 */

typedef vn-disjointness {
```

```
    type bits {
      bit node {
        position 0;
        description "node disjoint";
      }
      bit link {
        position 1;
        description "link disjoint";
      }
      bit srlg {
        position 2;
        description "srlg disjoint";
      }
    }
    description
      "type of the resource disjointness for
       VN level applied across all VN members
       in a VN";
  }

  grouping vn-ap {
    description
      "VNAP related information";
    leaf vn-ap-id {
      type uint32;
      description
        "unique identifier for the referred
         VNAP";
    }
    leaf vn {
      type leafref {
        path "/actn/vn/vn-list/vn-id";
      }
      description
        "reference to the VN";
    }
    leaf abstract-node {
      type leafref {
        path "/nw:networks/nw:network/nw:node/"
          + "tet:te-node-id";
      }
      description
        "a reference to the abstract node in TE
         Topology";
    }
  }
```

```
    leaf ltp {
        type te-types:te-tp-id;
        description
            "Reference LTP in the TE-topology";
    }
}
grouping access-point{
    description
        "AP related information";
    leaf access-point-id {
        type uint32;
        description
            "unique identifier for the referred
            access point";
    }
    leaf access-point-name {
        type string;
        description
            "ap name";
    }

    leaf max-bandwidth {
        type te-types:te-bandwidth;
        description
            "max bandwidth of the AP";
    }
    leaf avl-bandwidth {
        type te-types:te-bandwidth;
        description
            "available bandwidth of the AP";
    }
}
/*add details and any other properties of AP,
not associated by a VN
CE port, PE port etc.
*/
list vn-ap {
    key vn-ap-id;
    uses vn-ap;
    description
        "list of VNAP in this AP";
}
} //access-point
grouping vn-member {
    description
        "vn-member is described by this container";
```

```
leaf vn-member-id {
    type uint32;
    description
        "vn-member identifier";
}
container src
{
    description
        "the source of VN Member";
    leaf src {
        type leafref {
            path "/actn/ap/access-point-list/access-point-id";
        }
        description
            "reference to source AP";
    }
    leaf src-vn-ap-id{
        type leafref {
            path "/actn/ap/access-point-list/vn-ap/vn-ap-id";
        }
        description
            "reference to source VNAP";
    }
    leaf multi-src {
        if-feature multi-src-dest;
        type boolean;
        description
            "Is source part of multi-source, where
            only one of the source is enabled";
    }
}
container dest
{
    description
        "the destination of VN Member";
    leaf dest {
        type leafref {
            path "/actn/ap/access-point-list/access-point-id";
        }
        description
            "reference to destination AP";
    }
    leaf dest-vn-ap-id{
        type leafref {
            path "/actn/ap/access-point-list/vn-ap/vn-ap-id";
        }
    }
}
```

```
    }
    description
        "reference to dest VNAP";
    }
    leaf multi-dest {
        if-feature multi-src-dest;
        type boolean;
        description
            "Is destination part of multi-destination, where
            only one of the destination is enabled";
    }
}
leaf connetivity-matrix-id{
    type leafref {
        path "/nw:networks/nw:network/nw:node/tet:te/"
        + "tet:te-node-attributes/"
        + "tet:connectivity-matrices/"
        + "tet:connectivity-matrix/tet:id";
    }
    description
        "reference to connetivity-matrix";
}
} //vn-member
/*
grouping policy {
    description
        "policy related to vn-member-id";
    leaf local-reroute {
        type boolean;
        description
            "Policy to state if reroute
            can be done locally";
    }
    leaf push-allowed {
        type boolean;
        description
            "Policy to state if changes
            can be pushed to the customer";
    }
    leaf incremental-update {
        type boolean;
        description
            "Policy to allow only the
            changes to be reported";
    }
}
```

```
//policy
*/
grouping vn-policy {
  description
    "policy for VN-level diverisity";
  leaf vn-level-diversity {
    type vn-disjointness;
    description
      "the type of disjointness on the VN level
      (i.e., across all VN members)";
  }
}
/*
grouping metrics-op {
  description
    "metric related information";
  list metric{
    key "metric-type";
    config false;
    description
      "The list of metrics for VN";
    leaf metric-type {
      type identityref {
        base te-types:path-metric-type;
      }
      description
        "The VN metric type.";
    }
    leaf value{
      type uint32;
      description
        "The limit value";
    }
  }
}
/*
grouping metrics {
  description
    "metric related information";
  list metric{
    key "metric-type";
    description
      "The list of metrics for VN";
    uses te:path-metrics-bounds_config;
```

```

        container optimize{
            description
                "optimizing constraints";
            leaf enabled{
                type boolean;
                description
                    "Metric to optimize";
            }
            leaf value{
                type uint32;
                description
                    "The computed value";
            }
        }
    }
}
*/
/*
grouping service-metric {
    description
        "service-metric";
    uses te:path-objective-function_config;
    uses metrics;
    uses te-types:common-constraints_config;
    uses te:protection-restoration-params_config;
    uses policy;
} //service-metric
*/
/*
 * Configuration data nodes
 */
container actn {
    description
        "actn is described by this container";
    container ap {
        description
            "AP configurations";
        list access-point-list {
            key "access-point-id";
            description
                "access-point identifier";
            uses access-point{
                description
                    "access-point information";
            }
        }
    }
}

```



```
    }  
  }  
  container vn {  
    description  
      "VN configurations";  
    list vn-list {  
      key "vn-id";  
      description  
        "a virtual network is identified by a vn-id";  
      leaf vn-id {  
        type uint32;  
        description  
          "a unique vn identifier";  
      }  
      leaf vn-name {  
        type string;  
        description "vn name";  
      }  
      leaf vn-topology-id {  
        type te-types:te-topology-id;  
        description  
          "An optional identifier to the TE Topology  
          Model where the abstract nodes and links  
          of the Topology can be found for Type 2  
          VNS";  
      }  
      leaf abstract-node {  
        type leafref {  
          path "/nw:networks/nw:network/nw:node/"  
            + "tet:te-node-id";  
        }  
        description  
          "a reference to the abstract node in TE  
          Topology";  
      }  
      list vn-member-list {  
        key "vn-member-id";  
        description  
          "List of VN-members in a VN";  
        uses vn-member;  
        /*uses metrics-op;*/  
        leaf oper-status {  
          type identityref {  
            base vn-state-type;  
          }  
        }  
      }  
    }  
  }  
}
```

```
        config false;
        description
            "VN-member operational state.";
    }
}
leaf if-selected{
    if-feature multi-src-dest;
    type boolean;
    default false;
    config false;
    description
        "Is the vn-member is selected among the
        multi-src/dest options";
}
/*
container multi-src-dest{
    if-feature multi-src-dest;
    config false;
    description
        "The selected VN Member when multi-src
        and/or mult-destination is enabled.";
    leaf selected-vn-member{
        type leafref {
            path "/actn/vn/vn-list/vn-member-list"
                + "/vn-member-id";
        }
        description
            "The selected VN Member along the set
            of source and destination configured
            with multi-source and/or multi-destination";
    }
}
*/
/*uses service-metric;*/
leaf admin-status {
    type identityref {
        base vn-admin-state-type;
    }
    default vn-admin-state-up;
    description "VN administrative state.";
}
leaf oper-status {
    type identityref {
        base vn-state-type;
```

```

    }
    config false;
    description "VN operational state.";
  }
  uses vn-policy;
} //vn-list
} //vn
} //actn
/*
* Notifications - TBD
*/
/*
* RPC
*/
rpc vn-compute{
  description
    "The VN computation without actual
    instantiation";
  input {
    leaf abstract-node {
      type leafref {
        path "/nw:networks/nw:network/nw:node/"
          + "tet:te-node-id";
      }
      description
        "a reference to the abstract node in TE
        Topology";
    }
    list vn-member-list{
      key "vn-member-id";
      description
        "List of VN-members in a VN";
      uses vn-member;
    }
    uses vn-policy;
    /*uses service-metric;*/
  }
  output {
    list vn-member-list{
      key "vn-member-id";
      description
        "List of VN-members in a VN";
      uses vn-member;
      leaf if-selected{
        if-feature multi-src-dest;

```

```

        type boolean;
        default false;
        description
            "Is the vn-member is selected among
            the multi-src/dest options";
    }
    /*uses metrics-op;*/
    leaf compute-status {
        type identityref {
            base vn-compute-state-type;
        }
        description
            "VN-member compute state.";
    }
}
/*
container multi-src-dest{
    if-feature multi-src-dest;
    description
        "The selected VN Member when multi-src
        and/or mult-destination is enabled.";
    leaf selected-vn-member-id{
        type uint32;
        description
            "The selected VN Member-id from the
            input";
    }
}*/
}
}
}
}
}

```

<CODE ENDS>

7. JSON Example

This section provides json implementation examples as to how ACTN VN YANG model and TE topology model are used together to instantiate virtual networks.

The example in this section includes following VN

- o VN1 (Type 1): Which maps to the single node topology abstract1 (node D1) and consist of VN Members 104 (L1 to L4), 107 (L1 to L7), 204 (L2 to L4), 308 (L3 to L8) and 108 (L1 to L8). We also show how disjointness (node, link, srlg) is supported in the example on the global level (i.e., connectivity matrices level).
- o VN2 (Type 2): Which maps to the single node topology abstract2 (node D2), this topology has an underlay topology (absolute) (see figure in section 3.2). This VN has a single VN member 105 (L1 to L5) and an underlay path (S4 and S7) has been set in the connectivity matrix of abstract2 topology;
- o VN3 (Type 1): This VN has a multi-source, multi-destination feature enable for VN Member 104 (L1 to L4)/107 (L1 to L7) [multi-src] and VN Member 204 (L2 to L4)/304 (L3 to L4) [multi-dest] usecase. The selected VN-member is known via the field "if-selected" and the corresponding connectivity-matrix-id.

Note that the ACTN VN YANG model also include the AP and VNAP which shows various VN using the same AP.

7.1. ACTN VN JSON

```
{
  "actn":{
    "ap":{
      "access-point-list": [
        {
          "access-point-id": 101,
          "access-point-name": "101",
          "vn-ap": [
            {
              "vn-ap-id": 10101,
              "vn": 1,
              "abstract-node": "D1",
              "ltp": "1-0-1"
            },
            {
              "vn-ap-id": 10102,
              "vn": 2,
              "abstract-node": "D2",
              "ltp": "1-0-1"
            },
            {
              "vn-ap-id": 10103,
              "vn": 3,
              "abstract-node": "D3",

```

```

        "ltp": "1-0-1"
      },
    ],
  },
  {
    "access-point-id": 202,
    "access-point-name": "202",
    "vn-ap": [
      {
        "vn-ap-id": 20201,
        "vn": 1,
        "abstract-node": "D1",
        "ltp": "2-0-2"
      }
    ]
  },
  {
    "access-point-id": 303,
    "access-point-name": "303",
    "vn-ap": [
      {
        "vn-ap-id": 30301,
        "vn": 1,
        "abstract-node": "D1",
        "ltp": "3-0-3"
      },
      {
        "vn-ap-id": 30303,
        "vn": 3,
        "abstract-node": "D3",
        "ltp": "3-0-3"
      }
    ]
  },
  {
    "access-point-id": 440,
    "access-point-name": "440",
    "vn-ap": [
      {
        "vn-ap-id": 44001,
        "vn": 1,
        "abstract-node": "D1",
        "ltp": "4-4-0"
      }
    ]
  },
  {
    "access-point-id": 550,
    "access-point-name": "550",

```

```

        "vn-ap": [
            {
                "vn-ap-id": 55002,
                "vn": 2,
                "abstract-node": "D2",
                "ltp": "5-5-0"
            }
        ]
    },
    {
        "access-point-id": 770,
        "access-point-name": "770",
        "vn-ap": [
            {
                "vn-ap-id": 77001,
                "vn": 1,
                "abstract-node": "D1",
                "ltp": "7-7-0"
            },
            {
                "vn-ap-id": 77003,
                "vn": 3,
                "abstract-node": "D3",
                "ltp": "7-7-0"
            }
        ]
    },
    {
        "access-point-id": 880,
        "access-point-name": "880",
        "vn-ap": [
            {
                "vn-ap-id": 88001,
                "vn": 1,
                "abstract-node": "D1",
                "ltp": "8-8-0"
            },
            {
                "vn-ap-id": 88003,
                "vn": 3,
                "abstract-node": "D3",
                "ltp": "8-8-0"
            }
        ]
    }
]
},
"vn": {
    "vn-list": [

```

```
{
  "vn-id": 1,
  "vn-name": "vn1",
  "vn-topology-id": "te-topology:abstract1",
  "abstract-node": "D1",
  "vn-member-list": [
    {
      "vn-member-id": 104,
      "src": {
        "src": 101,
        "src-vn-ap-id": 10101,
      },
      "dest": {
        "dest": 440,
        "dest-vn-ap-id": 44001,
      },
      "connectivity-matrix-id": 104
    },
    {
      "vn-member-id": 107,
      "src": {
        "src": 101,
        "src-vn-ap-id": 10101,
      },
      "dest": {
        "dest": 770,
        "dest-vn-ap-id": 77001,
      },
      "connectivity-matrix-id": 107
    },
    {
      "vn-member-id": 204,
      "src": {
        "src": 202,
        "dest-vn-ap-id": 20401,
      },
      "dest": {
        "dest": 440,
        "dest-vn-ap-id": 44001,
      },
      "connectivity-matrix-id": 204
    },
    {
      "vn-member-id": 308,
      "src": {
        "src": 303,
        "src-vn-ap-id": 30301,
      },
      "dest": {
```



```

        "dest": 880,
        "src-vn-ap-id": 88001,
    },
    "connectivity-matrix-id": 308
  },
  {
    "vn-member-id": 108,
    "src": {
      "src": 101,
      "src-vn-ap-id": 10101,
    },
    "dest": {
      "dest": 880,
      "dest-vn-ap-id": 88001,
    },
    "connectivity-matrix-id": 108
  }
]
},
{
  "vn-id": 2,
  "vn-name": "vn2",
  "vn-topology-id": "te-topology:abstract2",
  "abstract-node": "D2",
  "vn-member-list": [
    {
      "vn-member-id": 105,
      "src": {
        "src": 101,
        "src-vn-ap-id": 10102,
      },
      "dest": {
        "dest": 550,
        "dest-vn-ap-id": 55002,
      },
      "connectivity-matrix-id": 105
    }
  ]
},
{
  "vn-id": 3,
  "vn-name": "vn3",
  "vn-topology-id": "te-topology:abstract3",
  "abstract-node": "D3",
  "vn-member-list": [
    {
      "vn-member-id": 104,
      "src": {
        "src": 101,

```

```

    },
    "dest": {
      "dest": 440,
      "multi-dest": true
    }
  },
  {
    "vn-member-id": 107,
    "src": {
      "src": 101,
      "src-vn-ap-id": 10103,
    },
    "dest": {
      "dest": 770,
      "dest-vn-ap-id": 77003,
      "multi-dest": true
    },
    "connectivity-matrix-id": 107,
    "if-selected": true,
  },
  {
    "vn-member-id": 204,
    "src": {
      "src": 202,
      "multi-src": true,
    },
    "dest": {
      "dest": 440,
    },
  },
  {
    "vn-member-id": 304,
    "src": {
      "src": 303,
      "src-vn-ap-id": 30303,
      "multi-src": true,
    },
    "dest": {
      "dest": 440,
      "src-vn-ap-id": 44003,
    },
    "connectivity-matrix-id": 304,
    "if-selected": true,
  },
],
},
]
}

```

```

    }
  }
}

```

7.2. TE-topology JSON

```

{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {}
        },
        "network-id": "abstract1",
        "provider-id": 201,
        "client-id": 600,
        "te-topology-id": "te-topology:abstract1",
        "node": [
          {
            "node-id": "D1",
            "te-node-id": "2.0.1.1",
            "te": {
              "te-node-attributes": {
                "domain-id" : 1,
                "is-abstract": [null],
                "connectivity-matrices": {
                  "is-allowed": true,
                  "path-constraints": {
                    "bandwidth-generic": {
                      "te-bandwidth": {
                        "generic": [
                          {
                            "generic": "0x1p10",
                          }
                        ]
                      }
                    }
                  }
                }
              },
              "disjointness": "node link srlg",
            },
            "connectivity-matrix": [
              {
                "id": 104,
                "from": "1-0-1",
                "to": "4-4-0"
              },
              {
                "id": 107,
                "from": "1-0-1",

```

```

        "to": "7-7-0"
      },
      {
        "id": 204,
        "from": "2-0-2",
        "to": "4-4-0"
      },
      {
        "id": 308,
        "from": "3-0-3",
        "to": "8-8-0"
      },
      {
        "id": 108,
        "from": "1-0-1",
        "to": "8-8-0"
      },
    ],
  },
}
},
"termination-point": [
  {
    "tp-id": "1-0-1",
    "te-tp-id": 10001,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-1-0",
    "te-tp-id": 10100,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "2-0-2",
    "te-tp-id": 20002,

```

```

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

```

```
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  },
  {
    "tp-id": "4-4-0",
    "te-tp-id": 40400,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "5-0-5",
    "te-tp-id": 50005,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "5-5-0",
    "te-tp-id": 50500,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "6-0-6",
    "te-tp-id": 60006,
    "te": {
      "interface-switching-capability": [
```

```
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "6-6-0",
    "te-tp-id": 60600,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "7-0-7",
    "te-tp-id": 70007,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "7-7-0",
    "te-tp-id": 70700,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "8-0-8",
    "te-tp-id": 80008,
    "te": {
      "interface-switching-capability": [
        {
```

```

        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
    }
}
],
},
{
    "tp-id": "8-8-0",
    "te-tp-id": 80800,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
}
]
}
},
],
},
},
{
    "network-types": {
        "te-topology": {}
    },
    "network-id": "abstract2",
    "provider-id": 201,
    "client-id": 600,
    "te-topology-id": "te-topology:abstract2",
    "node": [
        {
            "node-id": "D2",
            "te-node-id": "2.0.1.2",
            "te": {
                "te-node-attributes": {
                    "domain-id": 1,
                    "is-abstract": [null],
                    "connectivity-matrices": {
                        "is-allowed": true,
                        "underlay": {
                            "enabled": true
                        }
                    },
                    "path-constraints": {
                        "bandwidth-generic": {
                            "te-bandwidth": {
                                "generic": [
                                    {
                                        "generic": "0xlp10"

```



```

    }
  ]
}
},
"optimizations": {
  "objective-function": {
    "objective-function-type": "of-maximize-residual-
bandwidth"
  }
},
"connectivity-matrix": [
  {
    "id": 105,
    "from": "1-0-1",
    "to": "5-5-0",
    "underlay": {
      "enabled": true,
      "primary-path": {
        "network-ref": "absolute",
        "path-element": [
          {
            "path-element-id": 1,
            "index": 1,
            "numbered-hop": {
              "address": "4.4.4.4",
              "hop-type": "STRICT"
            }
          },
          {
            "path-element-id": 2,
            "index": 2,
            "numbered-hop": {
              "address": "7.7.7.7",
              "hop-type": "STRICT"
            }
          }
        ]
      }
    }
  }
]
}
},
"termination-point": [
  {
    "tp-id": "1-0-1",
    "te-tp-id": 10001,

```

```
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
},
{
  "tp-id": "1-1-0",
  "te-tp-id": 10100,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "2-0-2",
  "te-tp-id": 20002,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "2-2-0",
  "te-tp-id": 20200,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "3-0-3",
  "te-tp-id": 30003,
  "te": {
```

```
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  },
  {
    "tp-id": "3-3-0",
    "te-tp-id": 30300,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "4-0-4",
    "te-tp-id": 40004,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "4-4-0",
    "te-tp-id": 40400,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "5-0-5",
    "te-tp-id": 50005,
    "te": {
      "interface-switching-capability": [
```

```

        {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
        }
    ]
}
},
{
    "tp-id": "5-5-0",
    "te-tp-id": 50500,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "6-0-6",
    "te-tp-id": 60006,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "6-6-0",
    "te-tp-id": 60600,
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
            }
        ]
    }
},
{
    "tp-id": "7-0-7",
    "te-tp-id": 70007,
    "te": {
        "interface-switching-capability": [
            {

```

```

        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  },
  {
    "tp-id": "7-7-0",
    "te-tp-id": 70700,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "8-0-8",
    "te-tp-id": 80008,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "8-8-0",
    "te-tp-id": 80800,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
]
},
{
  "network-types": {
    "te-topology": {}
  }
}

```

```

    },
    "network-id": "abstract3",
    "provider-id": 201,
    "client-id": 600,
    "te-topology-id": "te-topology:abstract3",
    "node": [
      {
        "node-id": "D3",
        "te-node-id": "3.0.1.1",
        "te": {
          "te-node-attributes": {
            "domain-id" : 3,
            "is-abstract": [null],
            "connectivity-matrices": {
              "is-allowed": true,
              "path-constraints": {
                "bandwidth-generic": {
                  "te-bandwidth": {
                    "generic": [
                      {
                        "generic": "0x1p10",
                      }
                    ]
                  }
                }
              }
            },
            "connectivity-matrix": [
              {
                "id": 107,
                "from": "1-0-1",
                "to": "7-7-0"
              },
              {
                "id": 308,
                "from": "3-0-3",
                "to": "8-8-0"
              }
            ]
          }
        }
      }
    ],
    "termination-point": [
      {
        "tp-id": "1-0-1",
        "te-tp-id": 10001,
        "te": {
          "interface-switching-capability": [
            {
              "switching-capability": "switching-otn",
            }
          ]
        }
      }
    ]
  }

```

```

        "encoding": "lsp-encoding-oduk"
      }
    ]
  },
  {
    "tp-id": "1-1-0",
    "te-tp-id": 10100,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "2-0-2",
    "te-tp-id": 20002,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "2-2-0",
    "te-tp-id": 20200,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "3-0-3",
    "te-tp-id": 30003,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
]

```

```

    }
  ]
}
},
{
  "tp-id": "3-3-0",
  "te-tp-id": 30300,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "4-0-4",
  "te-tp-id": 40004,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "4-4-0",
  "te-tp-id": 40400,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "5-0-5",
  "te-tp-id": 50005,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}

```



```
    ]
  }
},
{
  "tp-id": "5-5-0",
  "te-tp-id": 50500,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "6-0-6",
  "te-tp-id": 60006,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "6-6-0",
  "te-tp-id": 60600,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "7-0-7",
  "te-tp-id": 70007,
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
```

```

    },
    {
      "tp-id": "7-7-0",
      "te-tp-id": 70700,
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    }
  ],
  {
    "tp-id": "8-0-8",
    "te-tp-id": 80008,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "8-8-0",
    "te-tp-id": 80800,
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
],
},
]
}

```

8. Security Considerations

TDB

9. IANA Considerations

TDB

10. Acknowledgments

The authors would like to thank Xufeng Liu for his helpful comments and valuable suggestions.

11. References

11.1. Normative References

- [TE-TOPO] X. Liu, et al., "YANG Data Model for TE Topologies", work in progress: draft-ietf-teas-yang-te-topo.
- [TE-tunnel] T. Saad, et al., "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", work in progress: draft-ietf-teas-yang-te.

11.2. Informative References

- [RFC7926] A. Farrel (Ed.), "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.
- [ACTN-REQ] Lee, et al., "Requirements for Abstraction and Control of TE Networks", draft-ietf-teas-actn-requirements, work in progress.
- [ACTN-FWK] D. Ceccarelli, Y. Lee [Editors], "Framework for Abstraction and Control of Traffic Engineered Networks", draft-ceccarelli-teas-actn-framework, work in progress.
- [TE-MAP] Y. Lee, D. Dhody, and D. Ceccarelli, "Traffic Engineering and Service Mapping Yang Model", draft-lee-teas-te-service-mapping-yang, work in progress.
- [SERVICE-YANG] Q. Wu, W. Liu and A. Farrel, "Service Models Explained", draft-wu-opsawg-service-model-explained, work in progress.
- [ACTN-PM] Y. Lee, et al., "YANG models for ACTN TE Performance Monitoring Telemetry and Network Autonomics", draft-lee-teas-actn-pm-telemetry-autonomics, work in progress.
- [OIF-VTNS] Virtual Transport Network Services 1.0 Specification, IA OIF-VTNS-1.0, April 2017.
- [L1CSM] G. Fioccola, Ed. & Y. Lee, Ed., "A Yang Data Model for L1 Connectivity Service Model (L1CSM)", draft-ietf-ccamp-l1csm-yang, work in progress.

- [L2SM] G. Fioccola, Ed., "A YANG Data Model for L2VPN Service Delivery", draft-ietf-l2sm-l2vpn-service-model, work in progress.
- [RFC8299] Q. Wu, Ed., S. Litkowski, L. Tomotaki, and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, January 2018.
- [RFC8309] Q. Wu, W. Cheng, and A. Farrel. "Service Models Explained", RFC 8309, January 2018.

12. Contributors

Contributor's Addresses

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

Xian Zhang
Huawei Technologies
Email: zhang.xian@huawei.com

Sergio Belotti
Nokia
Email: sergio.belotti@nokia.com

Takuya Miyasaka
KDDI
Email: ta-miyasaka@kddi.com

Authors' Addresses

Young Lee (ed.)
Huawei Technologies
Email: leeyoung@huawei.com

Dhruv Dhody
Huawei Technologies

Email: dhruv.ietf@gmail.com

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

Igor Bryskin
Huawei
Email: Igor.Bryskin@huawei.com

Bin Yeong Yoon
ETRI
Email: byyun@etri.re.kr

Qin Wu
Huawei Technologies
Email: bill.wu@huawei.com

Peter Park
KT
Email: peter.park@kt.com

TEAS Working Group
Internet-Draft
Intended status: Experimental
Expires: December 28, 2018

A. Wang
China Telecom
X. Huang
C. Kou
BUPT
Z. Li
China Mobile
L. Huang
P. Mi
Huawei Technologies
June 26, 2018

CCDR Scenario, Simulation and Suggestion
draft-ietf-teas-native-ip-scenarios-01

Abstract

This document describes the scenarios, simulation and suggestions for the "Centrally Control Dynamic Routing (CCDR)" architecture, which integrates the merit of traditional distributed protocols (IGP/BGP), and the power of centrally control technologies (PCE/SDN) to provide one feasible traffic engineering solution in various complex scenarios for the service provider.

Traditional MPLS-TE solution is mainly used in static network planning scenario and is difficult to meet the QoS assurance requirements in real-time traffic network. With the emerge of SDN concept and related technologies, it is possible to simplify the complexity of distributed control protocol, utilize the global view of network condition, give more efficient solution for traffic engineering in various complex scenarios.

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 December 28, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. Conventions used in this document	3
3. CCDR Scenarios.	3
3.1. Qos Assurance for Hybrid Cloud-based Application.	3
3.2. Increase link utilization based on tidal phenomena.	4
3.3. Traffic engineering for IDC/MAN asymmetric link	5
3.4. Network temporal congestion elimination.	6
4. CCDR Simulation.	6
4.1. Topology Simulation	6
4.2. Traffic Matrix Simulation.	7
4.3. CCDR End-to-End Path Optimization	7
4.4. Network temporal congestion elimination	9
5. CCDR Deployment Consideration.	10
6. Security Considerations	11
7. IANA Considerations	11
8. Normative References	11
Authors' Addresses	12

1. Introduction

Internet network is composed mainly tens of thousands of routers that run distributed protocol to exchange the reachability information between them. The path for the destination network is mainly calculated and controlled by the traditional IGP protocols. These distributed protocols are robust enough to support the current evolution of Internet but has some difficulties when the application requires the end-to-end QoS performance, or the service provider wants to maximize the links utilization within their network.

MPLS-TE technology is one perfect solution for the finely planned network but it will put heavy burden on the router when we use it to solve the dynamic QoS assurance requirements within real time traffic network.

SR(Segment Routing) is another prominent solution that integrates some merits of traditional distributed protocol and the advantages of centrally control mode, but it requires the underlying network, especially the provider edge router to do label push and pop action in-depth, and need some complex solutions for co-exist with the Non-SR network. Finally, it can only maneuver the end-to-end path for MPLS and IPv6 traffic via different mechanisms.

The advantage of MPLS is mainly for traffic isolation, such as the L2/L3 VPN service deployments, but most of the current application requirements are only for high performances end-to-end QoS assurance. Without the help of centrally control architecture, the service provider almost can't make such SLA guarantees upon the real time traffic situation.

This draft gives some scenarios that the centrally control dynamic routing (CCDR) architecture can easily solve, without adding more extra burdening on the router. It also gives the PCE algorithm results under the similar topology, traffic pattern and network size to illustrate the applicability of CCDR architecture. Finally, it gives some suggestions for the implementation and deployment of CCDR.

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. CCDR Scenarios.

The following sections describe some scenarios that the CCDR architecture is suitable for deployment.

3.1. Qos Assurance for Hybrid Cloud-based Application.

With the emerge of cloud computing technologies, enterprises are putting more and more services on the public oriented service infrastructure, but keep still some core services within their network. The bandwidth requirements between the private cloud and the public cloud are occasionally and the background traffic between these two sites varied from time to time. Enterprise cloud applications just want to invoke the network capabilities to make the

end-to-end QoS assurance on demand. Otherwise, the traffic should be controlled by the distributed protocol.

CCDR, which integrates the merits of distributed protocol and the power of centrally control, is suitable for this scenario. The possible solution architecture is illustrated below:

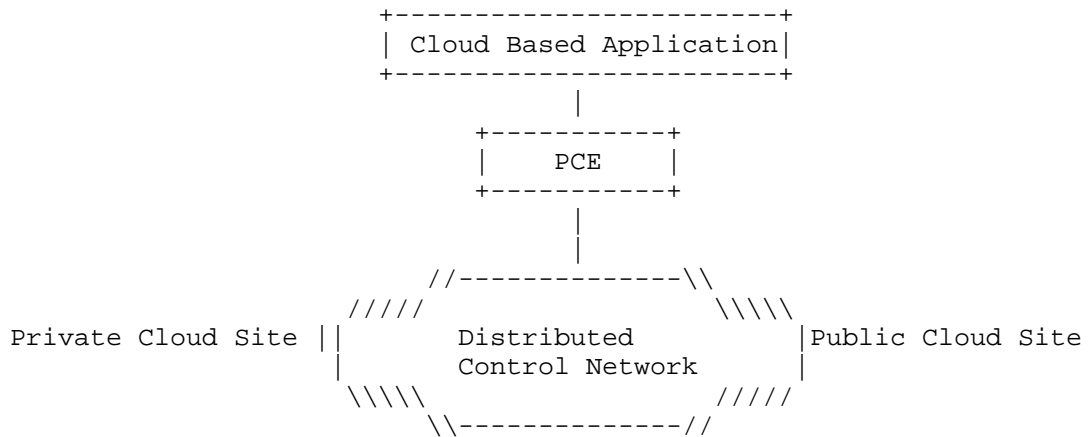


Fig.1 Hybrid Cloud Communication Scenario

By default, the traffic path between the private cloud site and public cloud site will be determined by the distributed control network. When some applications require the end-to-end QoS assurance, it can send these requirements to PCE, let PCE compute one e2e path which is based on the underlying network topology and the real traffic information, to accommodate the application's QoS requirements. The proposed solution can refer the draft [I-D.ietf-teas-pce-native-ip]. Section 4 describes the detail simulation process and the results.

3.2. Increase link utilization based on tidal phenomena.

Currently, the network topology within MAN is generally in star mode as illustrated in Fig.2, with the different devices connect different customer types. The traffic pattern of these customers demonstrates some tidal phenomena that the links between the CR/BRAS and CR/SR will experience congestion in different periods because the subscribers under BRAS often use the network at night and the dedicated line users under SR often use the network during the daytime. The uplink between BRAS/SR and CR must satisfy the maximum traffic pattern between them and this causes the links underutilization.

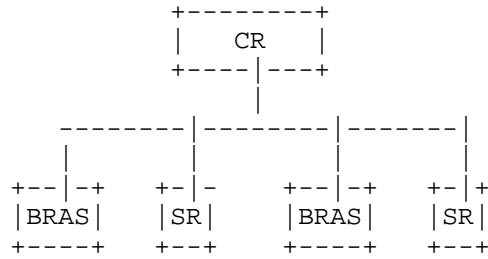


Fig.2 STAR-style network topology within MAN

If we can consider link the BRAS/SR with local loop, and control the MAN with the CCDR architecture, we can exploit the tidal phenomena between BRAS/CR and SR/CR links, increase the efficiency of them.

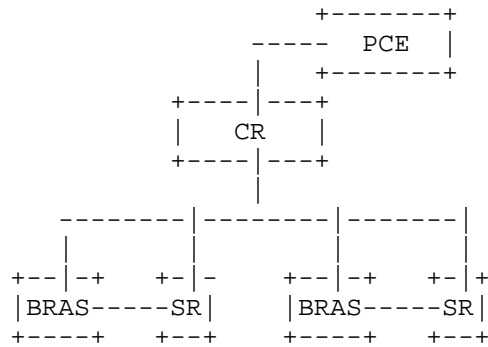


Fig.3 Increase the link utilization via CCDR

3.3. Traffic engineering for IDC/MAN asymmetric link

The operator's networks are often comprised by tens of different domains, interconnected with each other, form very complex topology that illustrated in Fig.4. Due to the traffic pattern to/from MAN and IDC, the links between them are often in asymmetric style. It is almost impossible to balance the utilization of these links via the distributed protocol, but this unbalance phenomenon can be overcome via the CCDR architecture.

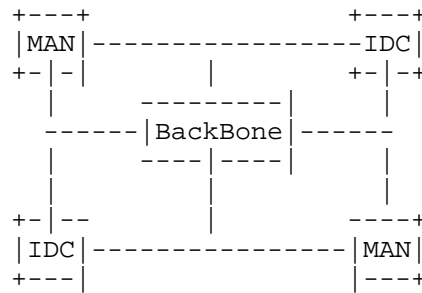


Fig.4 TE within Complex Multi-Domain topology

3.4. Network temporal congestion elimination.

In more general situation, there are often temporal congestion periods within part of the service provider's network. Such congestion phenomena will appear repeatedly and if the service provider has some methods to mitigate it, it will certainly increase the satisfaction degree of their customer. CCDR is also suitable for such scenario that the traditional distributed protocol will process most of the traffic forwarding and the controller will schedule some traffic out of the congestion links to lower the utilization of them. Section 4 describes the simulation process and results about such scenario.

4. CCDR Simulation.

The following sections describe the topology, traffic matrix, end-to-end path optimization and congestion elimination in CCDR simulation.

4.1. Topology Simulation

The network topology mainly contains nodes and links information. Nodes used in simulation have two types: core nodes and edge nodes. The core nodes are fully linked to each other. The edge nodes are connected only with some of the core nodes. Fig.5 is a topology example of 4 core nodes and 5 edge nodes. In CCDR simulation, 100 core nodes and 400 edge nodes are generated.

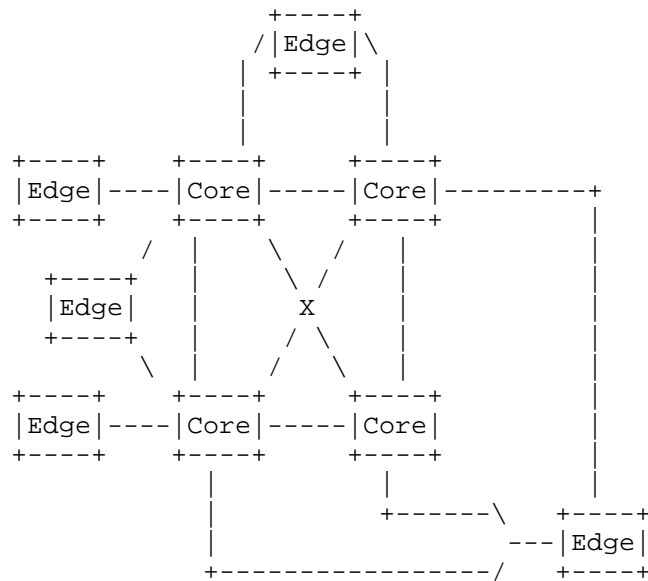


Fig.5 Topology of simulation

The number of links connecting one edge node to the set of core nodes is randomly between 2 to 30, and the total number of links is more than 20000. Each link has its congestion threshold.

4.2. Traffic Matrix Simulation.

The traffic matrix is generated based on the link capacity of topology. It can result in many kinds of situations, such as congestion, mild congestion and non-congestion.

In CCDD simulation, the traffic matrix is 500*500. About 20% links are overloaded when the Open Shortest Path First (OSPF) protocol is used in the network.

4.3. CCDR End-to-End Path Optimization

The CCDDR end-to-end path optimization is to find the best end-to-end path which is the lowest in metric value and each link of the path is far below link's threshold. Based on the current state of the network, PCE within CCDDR architecture combines the shortest path algorithm with penalty theory of classical optimization and graph theory.

Given background traffic matrix which is unscheduled, when a set of new flows comes into the network, the end-to-end path optimization finds the optimal paths for them. The selected paths bring the least congestion degree to the network.

The link utilization increment degree (UID) when the new flows are added into the network is shown in Fig.6. The first graph in Fig.6 is the UID with OSPF and the second graph is the UID with CCDR end-to-end path optimization. The average UID of graph one is more than 30%. After path optimization, the average UID is less than 5%. The results show that the CCDR end-to-end path optimization has an eye-catching decreasing in UID relative to the path chosen based on OSPF.

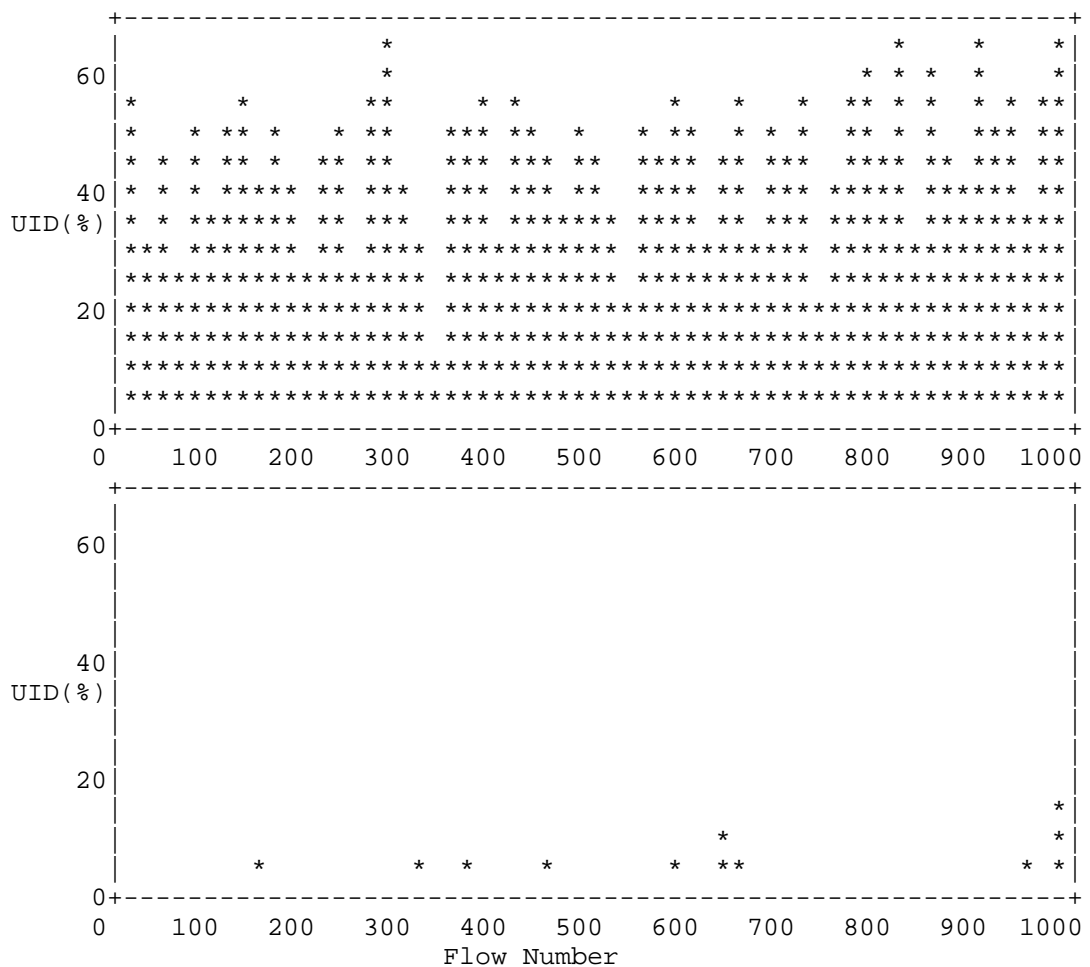


Fig.6 Simulation result with congestion elimination

4.4. Network temporal congestion elimination

Different degree of network congestion is simulated. The congestion degree (CD) is defined as the link utilization beyond its threshold.

The CCDR congestion elimination performance is shown in Fig.7. The first graph is the congestion degree before the process of congestion elimination. The average CD of all congested links is more than 10%. The second graph shown in Fig.7 is the congestion degree after congestion elimination process. It shows only 12 links among totally 20000 links exceed the threshold, and all the congestion degree is less than 3%. Thus, after schedule of the traffic in congestion paths, the degree of network congestion is greatly eliminated and the network utilization is in balance.

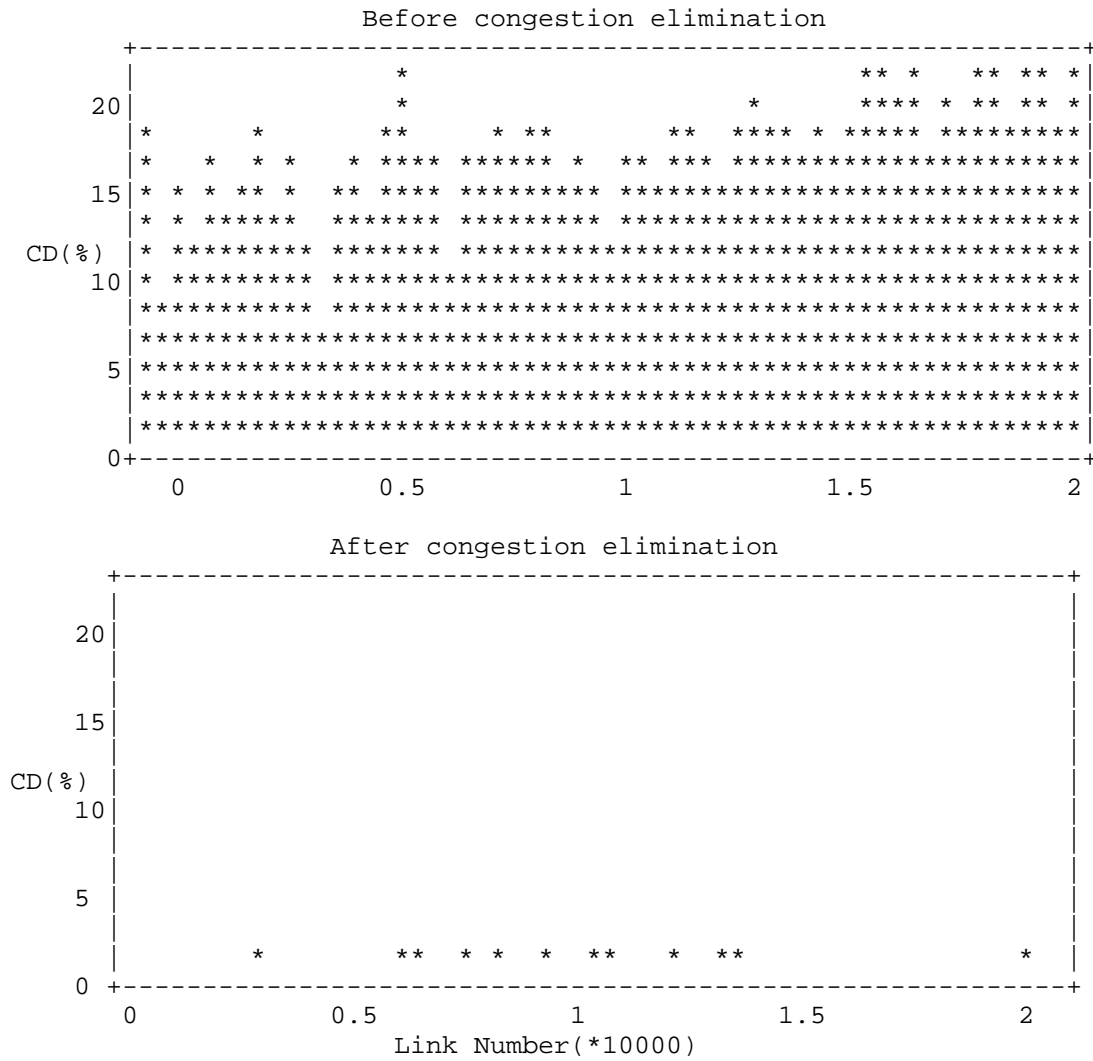


Fig.7 Simulation result with congestion elimination

5. CCDR Deployment Consideration.

With the above CCDR scenarios and simulation results, we can know it is necessary and feasible to find one general solution to cope with various complex situations for the most complex optimal path computation in centrally manner based on the underlay network topology and the real time traffic.

[I-D.ietf-teas-pce-native-ip] gives the principle solution for above scenarios, such thoughts can be extended to cover requirements that are more concretes in future.

6. Security Considerations

This document considers mainly the integration of traditional distributed protocol and the global view of central control. It certainly can ease the management of network in various traffic-engineering scenarios described in this document, but the central control manner may also bring the new point be easily attacked. Solutions for CCDR scenarios should keep these in mind and consider more for the protection of SDN controller and their communication with the underlay devices, which described in document 1 and [RFC8253]

7. IANA Considerations

This document does not require any IANA actions.

8. Normative References

- [I-D.ietf-teas-pce-native-ip]
Wang, A., Zhao, Q., Khasanov, B., and K. Mi, "PCE in Native IP Network", draft-ietf-teas-pce-native-ip-00 (work in progress), February 2018.
- [I-D.ietf-teas-pcecc-use-cases]
Zhao, Q., Li, Z., Khasanov, B., Ke, Z., Fang, L., Zhou, C., Communications, T., and A. Rachitskiy, "The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs", draft-ietf-teas-pcecc-use-cases-01 (work in progress), May 2017.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<https://www.rfc-editor.org/info/rfc5440>>.
- [RFC8253] Lopez, D., Gonzalez de Dios, O., Wu, Q., and D. Dhody, "PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)", RFC 8253, DOI 10.17487/RFC8253, October 2017, <<https://www.rfc-editor.org/info/rfc8253>>.

[RFC8283] Farrel, A., Ed., Zhao, Q., Ed., Li, Z., and C. Zhou, "An Architecture for Use of PCE and the PCE Communication Protocol (PCEP) in a Network with Central Control", RFC 8283, DOI 10.17487/RFC8283, December 2017, <<https://www.rfc-editor.org/info/rfc8283>>.

Authors' Addresses

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, Beijing 102209
China

Email: wangaj.bri@chinatelecom.cn

Xiaohong Huang
Beijing University of Posts and Telecommunications
No.10 Xitucheng Road, Haidian District
Beijing
China

Email: huangxh@bupt.edu.cn

Caixia Kou
Beijing University of Posts and Telecommunications
No.10 Xitucheng Road, Haidian District
Beijing
China

Email: koucx@lsec.cc.ac.cn

Zhenqiang Li
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing 100053
China

Email: li_zhenqiang@hotmail.com

Lu Huang
Huawei Technologies
Unit 7 NO 8.XiBinHe Road,YongDingMen
Beijing, Dongcheng District 100077
China

Email: hlisname@yahoo.com

Penghui Mi
Huawei Technologies
Tower C of Bldg.2, Cloud Park, No.2013 of Xuegang Road
Shenzhen, Bantian,Longgang District 518129
China

Email: mipenghui@huawei.com

TEAS Working Group
Internet-Draft
Intended status: Experimental
Expires: December 28, 2018

A. Wang
China Telecom
Q. Zhao
B. Khasanov
H. Chen
P. Mi
Huawei Technologies
R. Mallya
Juniper Networks
S. Peng
ZTE Corporation
June 26, 2018

PCE in Native IP Network
draft-ietf-teas-pce-native-ip-01

Abstract

This document defines the framework for CCN traffic engineering within Native IP network, using Dual/Multi-BGP session strategy and PCE-based central control architecture. The proposed central mode control framework conforms to the concept that defined in [RFC8283]. The scenario and simulation results of CCN traffic engineering is described in draft [I-D.ietf-teas-native-ip-scenarios].

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 December 28, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. Conventions used in this document	3
3. Dual-BGP framework for simple topology	3
4. Dual-BGP in large Scale Topology	4
5. Multi-BGP for Extended Traffic Differentiation	5
6. CCDR based framework for Multi-BGP strategy deployment	6
7. PCEP extension for key parameters delivery	7
8. Deployment Consideration	7
8.1. Scalability	8
8.2. High Availability	8
8.3. Incremental deployment	8
9. Security Considerations	8
10. IANA Considerations	9
11. Normative References	9
Authors' Addresses	10

1. Introduction

Draft [I-D.ietf-teas-native-ip-scenarios] describes the scenario and simulation results for the CCDR traffic engineering. In summary, the requirements for CCDR traffic engineering in Native IP network are the following:

- o No complex MPLS signaling procedure.
- o End to End traffic assurance, determined QoS behavior.
- o Identical deployment method for intra- and inter- domain.
- o No influence to existing router forward behavior.
- o Can utilize the power of centrally control(PCE) and flexibility/robustness of distributed control protocol.
- o Coping with the differentiation requirements for large amount traffic and prefixes.

- o Flexible deployment and automation control.

This document defines the framework for CCDR traffic engineering within Native IP network, using Dual/Multi-BGP session strategy and CCDR architecture, to meet the above requirements in dynamical and central control mode. Future PCEP protocol extensions to transfer the key parameters between PCE and the underlying network devices(PCC) are provided in draft [I-D.ietf-pce-pcep-extension-native-ip].

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. Dual-BGP framework for simple topology

Dual-BGP framework for simple topology is illustrated in Fig.1, which is comprised by SW1, SW2, R1, R2. There are multiple physical links between R1 and R2. Traffic between IP11 and IP21 is normal traffic, traffic between IP12 and IP22 is priority traffic that should be treated differently.

Only Native IGP/BGP protocol is deployed between R1 and R2. The traffic between each address pair may change timely and the corresponding source/destination addresses of the traffic may also change dynamically.

The key idea of the Dual-BGP framework for this simple topology is the following:

- o Build two BGP sessions between R1 and R2, via the different loopback address lo0, lo1 on these routers.
- o Send different prefixes via the two BGP sessions. (For example, IP11/IP21 via the BGP pair 1 and IP12/IP22 via the BGP pair 2).
- o Set the explicit peer route on R1 and R2 respectively for BGP next hop of lo0, lo1 to different physical link address between R1 and R2.

So, the traffic between the IP11 and IP21, and the traffic between IP12 and IP22 will go through different physical links between R1 and R2, each type of traffic occupy the different dedicated physical links.

If there is more traffic between IP12 and IP22 that needs to be assured , one can add more physical links on R1 and R2 to reach the loopback address 101(also the next hop for BGP Peer pair2). In this cases the prefixes that advertised by two BGP peer need not be changed.

If, for example, there is traffic from another address pair that needs to be assured (for example IP13/IP23), but the total volume of assured traffic does not exceed the capacity of the previous appointed physical links, then one need only to advertise the newly added source/destination prefixes via the BGP peer pair2, then the traffic between IP13/IP23 will go through the assigned dedicated physical links as the traffic between IP12/IP22.

Such decouple philosophy gives the network operator more flexible control ability on the network traffic, get the determined QoS assurance effect to meet the application's requirement. No complex MPLS signal procedures is introduced, the router need only support native IP protocol.

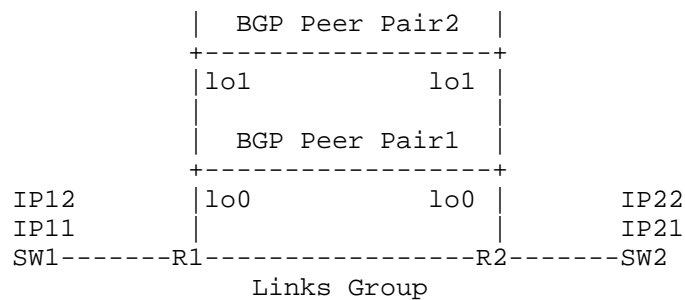


Fig.1 Design Philosophy for Dual-BGP Framework

4. Dual-BGP in large Scale Topology

When the assured traffic spans across one large scale network, as that illustrated in Fig.2, the dual BGP sessions cannot be established hop by hop especially for the iBGP within one AS.

For such scenario, we should consider to use the Route Reflector (RR) to achieve the similar Dual-BGP effect, select one router which performs the role of RR (for example R3 in Fig.2), every other edge router will establish two BGP peer sessions with the RR, using their different loopback addresses respectively. The other two steps for traffic differentiation are same as one described in the Dual-BGP simple topology usage case.

For the example shown in Fig.2, if we select the R1-R2-R4-R7 as the dedicated path, then we should set the explicit peer routes on these routers respectively, pointing to the BGP next hop (loopback addresses of R1 and R7, which are used to send the prefix of the assured traffic) to the actual address of the physical link.

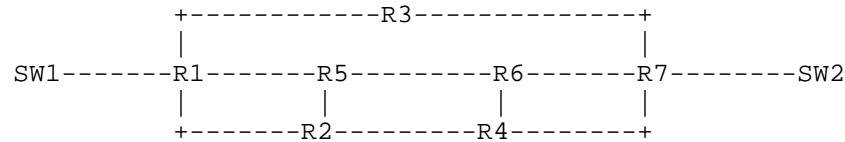


Fig.2 Dual-BGP Framework for large scale network

5. Multi-BGP for Extended Traffic Differentiation

In general situation, several additional traffic differentiation criteria exist, including:

- o Traffic that requires low latency links and is not sensitive to packet loss.
- o Traffic that requires low packet loss but can endure higher latency.
- o Traffic that requires lowest jitter path.
- o Traffic that requires high bandwidth links

These different traffic requirements can be summarized in the following table:

Flow No.	Latency	Packet Loss	Jitter
1	Low	Normal	Don't care
2	Normal	Low	Don't care
3	Normal	Normal	Low

Table 1. Traffic Requirement Criteria

For Flow No.1, we can select the shortest distance path to carry the traffic; for Flow No.2, we can select the idle links to form its end to end path; for Flow No.3, we can let all the traffic pass one single path, no ECMP distribution on the parallel links is required.

It is difficult and almost impossible to provide an end-to-end (E2E) path with latency, latency variation, packet loss, and bandwidth utilization constraints to meet the above requirements in large scale IP-based network via the traditional distributed routing protocol, but these requirements can be solved using the CCDR architecture since the PCE has the overall network view, can collect real network topology and network performance information about the underlying network, select the appropriate path to meet the various network performance requirements of different traffic type.

6. CCDR based framework for Multi-BGP strategy deployment

With the advent of SDN concepts towards pure IP networks, it is possible now to accomplish the central and dynamic control of network traffic according to the application's various requirements. The procedure to implement the dynamic deployment of Multi-BGP strategy is the following:

- o PCE gets topology and link utilization information from the underlying network, calculate the appropriate link path upon application's requirements..
- o PCE sends the key parameters to edge/RR routers(R1, R7 and R3 in Fig.3) to build multi-BGP peer relations and advertise different prefixes via them.
- o PCE sends the route information to the routers (R1,R2,R4,R7 in Fig.3) on forwarding path via PCEP, to build the path to the BGP next-hop of the advertised prefixes.
- o If the assured traffic prefixes were changed but the total volume of assured traffic does not exceed the physical capacity of the previous end-to-end path, then PCE needs only change the related information on edge routers (R1,R7 in Fig.3).
- o If volume of the assured traffic exceeds the capacity of previous calculated path, PCE must recalculate the appropriate path to accommodate the exceeding traffic via some new end-to-end physical link. After that PCE needs to update on-path routers to build such path hop by hop.

Fig.3 PCE based framework for Multi-BGP deployment

8.1. Scalability

In CCDR framework, PCE needs only to influence the edge routers for the prefixes differentiation via the multi-BGP deployment. The route information for these prefixes within the on-path routers were distributed via the traditional BGP protocol. Unlike the solution from BGP Flowspec, the on-path router need only keep the specific policy routes to the BGP next-hop of the differentiate prefixes, not the specific routes to the prefixes themselves. This can lessen the burden from the table size of policy based routes for the on-path routers, and has more scalability when comparing with the solution from BGP flowspec or Openflow.

8.2. High Availability

CCDR framework is based on the traditional distributed IP protocol. If the PCE failed, the forwarding plane will not be impacted, as the BGP session between all devices will not flap, and the forwarding table will remain the same. If one node on the optimal path is failed, the assurance traffic will fall over to the best-effort forwarding path. One can even design several assurance paths to load balance/hot standby the assurance traffic to meet the path failure situation, as done in MPLS FRR.

From PCE/SDN-controller HA side we will rely on existing HA solutions of SDN controllers such as clustering.

8.3. Incremental deployment

Not every router within the network support will support the PCEP extension that defined in [I-D.ietf-pce-pcep-extension-native-ip] simultaneously. For such situations, router on the edge of sub domain can be upgraded first, and then the traffic can be assured between different sub domains. Within each sub domain, the traffic will be forwarded along the best-effort path. Service provider can selectively upgrade the routers on each sub-domain in sequence.

9. Security Considerations

Solution described in this draft puts more requirements on the function of PCE and its communication with the underlay devices. The PCE should have the capability to calculate the loop-free e2e path upon the status of network condition and the service requirements in real time. The PCE need also to consider the router order during deployment to eliminate the possible transient traffic loop.

This solution does not require the change of forward behavior on the underlay devices, then there will no additional security impact for the devices.

When deploy the solution on network, service provider should also consider more on the protection of SDN controller and their communication with the underlay devices, which described in document [RFC5440] and [RFC8253]

10. IANA Considerations

This document does not require any IANA actions.

11. Normative References

- [I-D.ietf-pce-pcep-extension-native-ip]
Wang, A., Khasanov, B., Cheruathur, S., and C. Zhu, "PCEP Extension for Native IP Network", draft-ietf-pce-pcep-extension-native-ip-00 (work in progress), June 2018.
- [I-D.ietf-teas-native-ip-scenarios]
Wang, A., Huang, X., Qou, C., Huang, L., and K. Mi, "CCDR Scenario, Simulation and Suggestion", draft-ietf-teas-native-ip-scenarios-00 (work in progress), February 2018.
- [I-D.ietf-teas-pcecc-use-cases]
Zhao, Q., Li, Z., Khasanov, B., Ke, Z., Fang, L., Zhou, C., Communications, T., and A. Rachitskiy, "The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs", draft-ietf-teas-pcecc-use-cases-01 (work in progress), May 2017.
- [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>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<https://www.rfc-editor.org/info/rfc5440>>.
- [RFC8253] Lopez, D., Gonzalez de Dios, O., Wu, Q., and D. Dhody, "PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)", RFC 8253, DOI 10.17487/RFC8253, October 2017, <<https://www.rfc-editor.org/info/rfc8253>>.

[RFC8283] Farrel, A., Ed., Zhao, Q., Ed., Li, Z., and C. Zhou, "An Architecture for Use of PCE and the PCE Communication Protocol (PCEP) in a Network with Central Control", RFC 8283, DOI 10.17487/RFC8283, December 2017, <<https://www.rfc-editor.org/info/rfc8283>>.

Authors' Addresses

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, Beijing 102209
China

Email: wangaj.bri@chinatelecom.cn

Quintin Zhao
Huawei Technologies
125 Nagog Technology Park
Acton, MA 01719
USA

Email: quintin.zhao@huawei.com

Boris Khasanov
Huawei Technologies
Moskovskiy Prospekt 97A
St.Petersburg 196084
Russia

Email: khasanov.boris@huawei.com

Huaimo Chen
Huawei Technologies
Boston, MA
USA

Email: huaimo.chen@huawei.com

Penghui Mi
Huawei Technologies
Tower C of Bldg.2, Cloud Park, No.2013 of Xuegang Road
Bantian, Longgang District, Shenzhen 518129
China

Email: mipenghui@huawei.com

Raghavendra Mallya
Juniper Networks
1133 Innovation Way
Sunnyvale, California 94089
USA

Email: rmallya@juniper.net

Shaofu Peng
ZTE Corporation
No.68 Zijinghua Road, Yuhuatai District
Nanjing 210012
China

Email: peng.shaofu@zte.com.cn

TEAS WG
Internet-Draft
Intended status: Standards Track
Expires: December 30, 2018

A. Deshmukh
K. Kompella
Juniper Networks, Inc.
June 28, 2018

RSVP Extensions for RMR
draft-ietf-teas-rsvp-rmr-extension-01

Abstract

Ring topology is commonly found in access and aggregation networks. However, the use of MPLS as the transport protocol for rings is very limited today. draft-ietf-mpls-rmr-02 describes a mechanism to handle rings efficiently using MPLS. This document describes the extensions to the RSVP-TE protocol for signaling MPLS label switched paths in rings.

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 December 30, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Terminology	3
3. RSVP Extensions	4
3.1. Session Object	4
3.2. SENDER_TEMPLATE, FILTER_SPEC Objects	5
4. Ring Signaling Procedures	5
4.1. Differences from regular RSVP-TE LSPs	5
4.2. LSP signaling	6
4.2.1. Path Propagation for RMR	8
4.2.2. Resv Processing for RMR	8
4.3. Protection	9
4.4. Ring changes	10
4.5. Express Links	11
4.6. Bandwidth management	11
5. Security Considerations	13
6. Contributors	13
7. IANA Considerations	14
8. References	14
8.1. Normative References	14
8.2. Informative References	14
Authors' Addresses	15

1. Introduction

This document extends RSVP-TE [RFC3209] to establish label-switched path (LSP) tunnels in the ring topology. Rings are auto-discovered using the mechanisms mentioned in the [draft-ietf-mpls-rmr-02]. Either IS-IS [RFC5305] or OSPF[RFC3630] can be used as the IGP for auto-discovering the rings.

After the rings are auto-discovered, each node in the ring knows its clockwise(CW) and anti-clockwise (AC) ring neighbors and its ring links. All of the express links in the ring also get identified as part of the auto-discovery process. At this point, every node in the ring informs the RSVP protocol to begin the signaling of the ring LSPs.

Section 2 covers the terminology used in this document. Section 3 presents the RSVP protocol extensions needed to support MPLS rings. Section 4 describes the procedures of RSVP LSP signaling in detail.

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.

2. Terminology

Assuming there are n nodes in the network, a ring gets formed by a subset of those n nodes $\{R_i, R_{i+1}, R_{i+2}, \dots, R_n\}$. We define the direction from node R_i to R_{i+1} as "clockwise" (CW) and the reverse direction as "anti-clockwise" (AC). As there might be several rings in a graph, each ring is identified by its own distinct ring ID - RID.

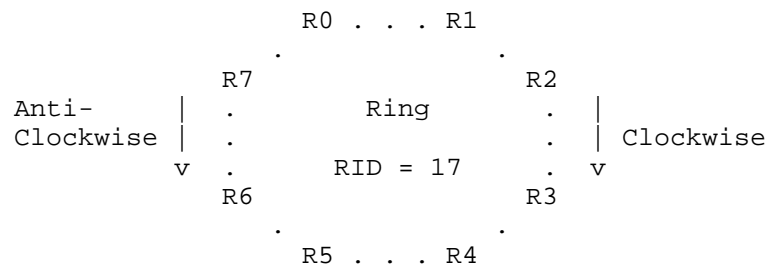


Figure 1: Ring with 8 nodes

The following terminology is used for ring LSPs:

Ring ID (RID): A non-zero number that identifies a ring; this is unique in a Service Provider's network. A node may belong to multiple rings.

Ring node: A member of a ring. Note that a device may belong to several rings.

Node index: A logical numbering of nodes in a ring, from zero up to one less than the ring size. Used purely for exposition in this document.

Ring neighbors: Nodes whose indices differ by one (modulo ring size).

Ring links: Links that connect ring neighbors.

Express links: Links that connect non-neighboring ring nodes.

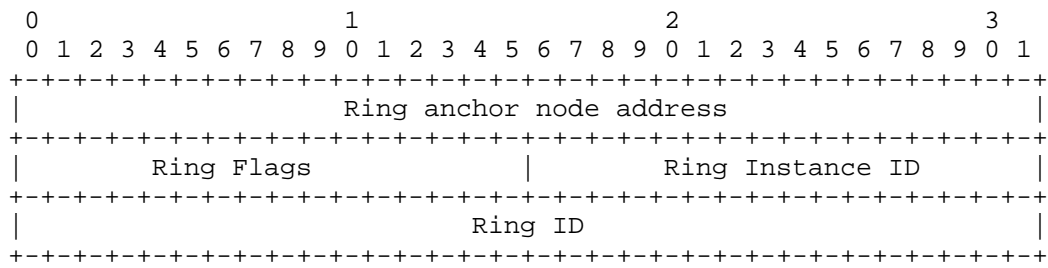
MP2P LSP: Each LSP in the ring is a multipoint to point LSP such that LSP can have multiple ingress nodes and one egress node.

3. RSVP Extensions

Due to the new ring LSP semantics, the signaling-message identification of ring LSPs will be different than the regular RSVP LSPs. So, a new C-Type is defined here for the SESSION object. This new C-Type will help to clearly differentiate ring LSPs from regular LSPs. In addition, new flags are introduced in the SESSION object to represent the ring direction of the corresponding Path message.

3.1. Session Object

Class = SESSION, LSP_TUNNEL_IPv4 C-Type = TBD



SESSION Object

Ring anchor node address: IPv4 address of the anchor node. Each anchor node creates a LSP addressed to itself.

Ring Instance ID: A 16-bit identifier used in the SESSION. This Ring Instance ID is useful for graceful ring changes. If a new node is being added to the ring(resulting in signaling of a larger ring) or some existing node goes down(resulting in signaling of a smaller ring), in those cases, anchor node creates a new tunnel with a different Ring Instance ID.

Ring ID: A 32-bit number that identifies a ring; this is unique in some scope of a Service Provider's network. This number remains constant throughout the existence of ring.

Ring Flags: For each ring, the anchor node starts signaling of a ring LSP. Ring LSP named RL_i , anchored on node R_i , consists of two counter-rotating unicast LSPs that start and end at R_i . One LSP will be in the clockwise direction and other LSP will be in

the anti-clockwise direction. A ring LSP is "multipoint": any node along the ring can use LSP RLi to send traffic to Ri; this can be in either the CW or AC directions, or both (i.e., load balanced). Two new flags are defined in the SESSION object which define the ring direction of the corresponding Path message.

ClockWise(CW) Direction 0x01: This flag indicates that the corresponding Path message is traveling in the ClockWise(CW) direction along the ring.

Anti-ClockWise(AC) Direction 0x02: This flag indicates that the corresponding Path message is traveling in the Anti-ClockWise(AC) direction along the ring.

3.2. SENDER_TEMPLATE, FILTER_SPEC Objects

There will be no changes to the SENDER_TEMPLATE and FILTER_SPEC objects. The format of the above 2 objects will be similar to the definitions in RFC 3209. [RFC3209] Only the semantics of these objects will slightly change. This will be explained in section Section 4.6 below.

4. Ring Signaling Procedures

A ring node indicates in its IGP updates the ring LSP signaling protocols that it supports. This can be LDP and/or RSVP-TE. Ideally, each node should support both. If the ring is configured with RSVP as the signaling protocol, then once a ring node R_i knows the RID, its ring links and directions, it kicks off ring RSVP LSP signaling automatically.

4.1. Differences from regular RSVP-TE LSPs

Ring LSPs differ from regular RSVP-TE LSPs in several ways:

1. Ring LSPs (by construction) form a loop.
2. Ring LSPs are multipoint-to-point. Any ring node can inject traffic into a ring LSP.
3. The bandwidth of a ring LSP can change hop-by-hop.
4. Ring LSPs are protected without the use of bypass or detour LSPs. Protection is handled by the ring LSP traversing in the opposite direction.

4.2. LSP signaling

After the ring auto-discovery process, each anchor node creates a LSP addressed to itself. This ring LSP contains of a pair of counter-rotating unicast LSPs. So, for a ring containing N nodes, there will be 2N total LSPs signaled.

There is no need for ERO object in the Path message. The Path message for ring LSPs has the following format:

```

<Path Message> ::= <Common Header> [ <INTEGRITY> ]
                        <SESSION> <RSVP_HOP>
                        <TIME_VALUES>
                        <LABEL_REQUEST>
                        [ <SESSION_ATTRIBUTE> ]
                        <sender descriptor list>

<sender descriptor list> ::= <sender descriptor>|
                                <sender descriptor list> <sender descri
ptor>

<sender descriptor> ::= <SENDER_TEMPLATE> <SENDER_TSPEC>

```

The anchor node creates 2 Path messages traveling in opposite directions. The SESSION format MUST be as per the description in Section 3.1. The anchor node which creates the LSP will insert it's own address in the "Ring anchor node address" field of the SESSION object. So effectively, the Path messages are addressed to the originating node itself.

The SESSION flags of these 2 Path messages are different. The Path message sent to the CW neighbor MUST have the CW flag set in the SESSION object to signal the LSP going in the clockwise direction. The Path message sent to the AC neighbor MUST have the AC flag set to signal the LSP in the anti-clockwise direction.

When an incoming Path message is received at the ring node Ri, it consults the results of auto-discovery to find the appropriate ring neighbor. If the incoming Path message has CW direction flag set, then Ri includes its own SENDER_DESCRIPTOR in the path message and forwards the Path message to its CW ring neighbor(Ri+1). Similarly if the incoming Path message has AC direction flag set, then Ri includes its own SENDER_TEMPLATE and forwards that Path message to it's AC ring neighbor(Ri-1). Thus, there is no need of ERO in the Path message. The Path message is routed locally at each ring based on the ring auto-discovery calculations.

The RESV message for ring LSPs also uses the new RING_IPv4 SESSION object. When the Path message originated from the anchor node R_i reaches back to R_i , R_i generates a Resv message. Note that this means that anchor node is both Ingress and Egress for the Path message. The Resv message copies the same ring flags as received in the corresponding Path message. So, a Resv message for a CW LSP goes in the AC direction (unlike the Path message, which goes CW). This is done to correctly match Path and corresponding Resv messages at transit ring nodes. Upon receiving Resv message with CW flag set, the ring node will forward the Resv message to its AC neighbor.

Each ring node R_i allocates CW and AC labels for each ring LSP $RL_x(x$ between $i..n$). As the signaling propagates around the ring, CW and AC labels are exchanged. When R_i receives CW and AC labels for LSP RL_x from its ring neighbors, primary and fast reroute (FRR) paths for RL_x are installed at R_i .

Consider the following three nodes of the ring, and their signaling interactions for LSP RL_5 originating from anchor node R_5 :

```

                P5_CW ->      P5_CW ->
                Q5_CW <-      Q5_CW <-
... ----- R7 ----- R8 ----- R9 ----- ...
                P5_AC <-      P5_AC <-
                Q5_AC ->      Q5_AC ->

```

P corresponds to the Path message and Q corresponds to the Resv message.

As explained above, an RMR LSP consists of two counter-rotating ring LSPs that start and end at the same node, say R_1 . As such, this appears to cause a loop, something that is normally avoided by RSVP-TE. There are some benefits to this:

Having a ring LSP form a loop allows the anchor node R_1 to ping itself and thus verify the end-to-end operation of the LSP. This, in conjunction with link-level OAM, offers a good indication of the operational state of the LSP. Also, having R_1 to be the ingress means that R_1 can initiate the Path messages for the two ring LSPs. This avoids R_1 having to coordinate with its neighbors to signal the LSPs, and simplifies the case where a ring update changes R_1 's ring neighbors. The cost of this is a little more signaling and a couple more label entries in the LFIB. However, we will let experiences from implementation guide us when we evaluate this approach.

4.2.1. Path Propagation for RMR

Ring LSPs are MP2P in nature. It means that every non-egress node is also an ingress and a merge-point for the LSP. Focussing on ring-LSP-0 (i.e ring-LSPs starting at R0):

```
R0---->R1---->R2---->R3---->R4---->R5---->R6--->R7--->R0(CW LSP)
R0---->R7---->R6---->R5---->R4---->R3---->R2--->R1--->R0(ACW LSP)
```

Each ring node inserts a new SENDER_TEMPLATE object into an incoming Path message. The procedure for that is as follows:

When a ring node R3 receives a Path message initiated by anchor node R0(for anchor lsp "lsp0"), R3 SHOULD make a copy of the received Path message for "lsp0". R3 then inserts a new sender-template object into the Path message for "lsp0". In the sender-template object, R3 uses the sender address as the loopback address of node R3 and lsp-id = X. R3 then forwards this modified Path message to it's ring neighbor.

So at this point, when Path messages heads out at R3, there will be 4 different SENDER_TEMPLATE objects in the outgoing Path message for lsp0:

```
-----
|SENDER_TEMPLATE_0 : SENDER_ADDRESS = R0, LSP_ID = 1 |
-----
|SENDER_TEMPLATE_1 : SENDER_ADDRESS = R1, LSP_ID = 1 |
-----
|SENDER_TEMPLATE_2 : SENDER_ADDRESS = R2, LSP_ID = 1 |
-----
|SENDER_TEMPLATE_3 : SENDER_ADDRESS = R3, LSP_ID = 1 |
-----
```

4.2.2. Resv Processing for RMR

When Egress node R0 receives the modified Path message, it replies with the a Resv message containing multiple FLOW_DESCRIPTOR objects. There should be 1 FLOW_DESCRIPTOR object corresponding to each of the SENDER_TEMPLATE object in the incoming Path message. The SESSION object of the Resv message will exactly match with the received Path message.

[RFC 3209] already supports receiving a Resv message with multiple flow-descriptors in it, as described in section 3.2 in that document. In each flow-descriptor there is a separate:

- a. FLOW_SPEC object corresponding to the SENDER_TSPEC that was sent in the Path message which could be admitted after admission-control downstream, and
- b. FILTER_SPEC object corresponding to SENDER_TEMPLATE that was sent in the Path message that could be admitted after admission-control downstream.

Each transit node removes the FLOW-DESCRIPTOR corresponding to itself from the Resv message before sending the Resv message upstream.

4.3. Protection

In the rings, there are no protection LSPs -- no node or link bypass LSPs, no standby LSPs and no detours. Protection is via the "other" direction around the ring, which is why ring LSPs are in counter-rotating pairs. Protection works in the same way for link, node and ring LSP failures.

Since each ring LSP is a MP2P LSP, any ring node can inject traffic onto a LSP whose anchor might be a different ring node. To achieve the above, an ingress route will be installed as follows at every ring node J, for a given ring-LSP with anchor Rk (say 1.2.3.4).

```
1.2.3.4  -> (Push CL_J+1,K, NH: R_J+1)      # CW
          -> (Push AL_J-1,K, NH: R_J-1)      # AC

CL = Clockwise label
AL = Anti-Clockwise label
```

Traffic will either be load balanced in the CW and AC directions or the traffic will be sent on just CW or AC lsp based on parameters such as hop-count, policy etc.

Also, 2 transit routes will be installed for the anchor LSP transiting from node Rj as follows:


```

CL_J,K -> SWAP(CL_J+1,K,  NH: R_J+1)      #CW
        -> SWAP(AL_J-1,K , NH: R_J-1)      #AC

CL = Clockwise label
AL = Anti-Clockwise label
CW NH has weight 1, AC NH has higher-weight.

AL_J,K -> SWAP(AL_J-1,K , NH: R_J-1)  #AC
        -> SWAP(CL_J+1,K,  NH: R_J+1)  #CW

CL = Clockwise label
AL = Anti-Clockwise label
AC NH has weight 1, CW NH has higher weight.

```

Suppose a packet headed in anti-clockwise direction towards R5 and it arrives at node R7. Lets say that now R7 learns there is a link failure in the AC direction. R7 reroutes this packet back onto the clockwise direction. This reroute action is pre-programmed in the LFIB, to minimize the time between detection of a fault and the corresponding recovery action.

At this time, R7 also sends a notification to R0 that the AC direction is not working. R0 modifies it's ingress route(for R5 LSP) by removing the AC direction LSP's route. Thus, R0 switches traffic to the CW direction.

These notification propagate CW until each traffic source on the ring CW of the failure uses the CW direction. For RSVP-TE, this notification is sent in the form of PathErr message.

To provide this notification, the ring node detecting failure SHOULD send a Path Error message with error code of "Notify" and an error value field of ("Tunnel locally repaired"). This Path Error code and value is same as defined in RFC 4090[RFC4090] for the notification of local repair.

Note that the failure of a node or a link will not necessarily affect all ring LSPs. Thus, it is important to identify the affected LSPs and only switch the affected LSPs.

4.4. Ring changes

A ring node can go down resulting in a smaller ring or a new node can be added to the ring which will increase the ring size. In both of

the above cases, the ring auto-discovery process SHOULD kick in and it SHOULD calculate a new ring with the changed ring nodes.

When the ring auto-discovery process is complete, IGP will signal RSVP to begin the MBB process for the existing ring LSPs. For this MBB process, the anchor node will create a new Path message with a different Ring Instance ID in the SESSION object. All other fields in the SESSION Object will remain same as the existing Path message(before the ring change).

This new Path message will then propagate along the ring neighbors in the same way as the original Path message. Each ring neighbor SHOULD forward the Path message to it's appropriate neighbor based on the new auto-discovery calculations.

For the ring links which are common between the old and new LSPs, the LSPs will share resources(SE style reservation) on those ring links. Note that here we are using Ring Instance ID in the SESSION object to share resources instead of the LSP_ID in the SENDER_TEMPLATE Object(which is used in RSVP-TE for sharing resources as described in RFC 3209 [RFC4090]). The LSP_ID use is reserved for a different functionality as described in section Section 4.6.

4.5. Express Links

The details for signaling over express links will be given in a future version.

4.6. Bandwidth management

For RSVP-TE LSPs, bandwidths may be signaled in both directions. However, these are not provisioned either; rather, one does "reverse call admission control". When a service needs to use an LSP, the ring node where the traffic enters the ring attempts to increase the bandwidth on the LSP to the egress. If successful, the service is admitted to the ring.

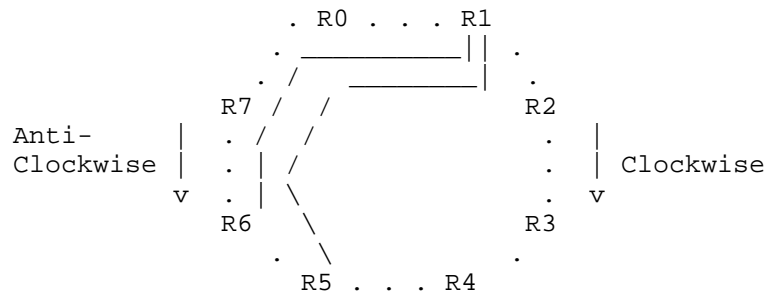


Figure 2: BW Management in Ring with 8 nodes

Let's say that Ring node R5 wants to increase the BW for the LSP whose egress is at node R1. To achieve this BW increase, Ring node R5 has to increase BW along the LSP anchored at node R1 (say lsp1).

R5 makes a copy of the existing ring Path message for lsp1. R5 then modifies the sender-template object from the copied Path message for "lsp1". In the sender-template object, R5 uses the sender address as the loopback address of node R5 and lsp-id = X+1. R5 also modifies the TSPEC object which represents the BW increase/decrease in this new Path message. R5 then forwards this new Path message to its ring neighbor. The original anchor Path message has sender address as loopback address of R1.

Now, let's say, node 5 wants to increase BW again for lsp1, then R5 adds a new SENDER_TEMPLATE object in the existing Path message for "lsp1" with sender address as loopback of node 5 and lsp-id = X+2. So at this point, there will be 2 different SENDER_TEMPLATE objects corresponding to node 5 in the outgoing path message.

```

-----
| SENDER_TEMPLATE_0 : SENDER_ADDRESS = R0, LSP_ID = 1 |
-----
| SENDER_TEMPLATE_1 : SENDER_ADDRESS = R1, LSP_ID = 1 |
-----
| ..... |
-----
| SENDER_TEMPLATE_5 : SENDER_ADDRESS = R5, LSP_ID = 1 |
-----
| SENDER_TEMPLATE_5 : SENDER_ADDRESS = R5, LSP_ID = 2 |
-----

```

Similarly, if node R6 wants to increase the BW for "lsp1", it SHOULD create a new Path message containing SENDER_TEMPLATE object with sender address = loopback of node 6 and lsp-id = Y+1. Thus, it

should be noted that each ring-node independently tracks its own lsp-ID that is currently in-use on a given RMR sub-LSP. This lsp-ID value will (could) be different for each ring-node for a given ring sub-LSP.

If sufficient BW is available all the way towards ring node R1, then this new Path message reaches node R1. R1 generates a Resv message with the correct FILTER_SPEC object corresponding to the received SENDER_TEMPLATE object. This Resv message will also have the correct FLOWSPEC object as per the requested bandwidth.

If sufficient BW is not available at some downstream (say node R9), then ring node R9 SHOULD generate a PathErr message with the corresponding Sender Template Object. When node R5 receives this PathErr message, R5 understands that the BW increase was not successful. Note that the existing established bandwidths for lsp1 are not affected by this new PathErr message.

When ring node R5 no longer needs the BW reservation, then ring node R5 SHOULD originate a new Path message with the appropriate Sender Template Object containing 0 BW as described above. Every downstream node SHOULD then remove bandwidth allocated on the corresponding link on receipt of this Path message.

Also, note that as part of this BW increase or decrease process, any ring node does not actually change any label associated with the LSP. So, the label remains same as it was signaled initially when the anchor LSP came up.

5. Security Considerations

It is not anticipated that either the notion of MPLS rings or the extensions to various protocols to support them will cause new security loopholes. As this document is updated, this section will also be updated.

6. Contributors

Ravi Singh
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
USA

Email: ravis@juniper.net

Santosh Esale
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
USA

Email: sesale@juniper.net

Raveendra Torvi
Juniper Networks, Inc.
10 Technology Park Dr
Westford, MA 01886
USA

Email: rtorvi@juniper.net

7. IANA Considerations

Requests to IANA will be made in a future version of this document.

8. References

8.1. Normative References

- [I-D.ietf-mpls-rmr]
Kompella, K. and L. Contreras, "Resilient MPLS Rings",
draft-ietf-mpls-rmr-07 (work in progress), March 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>>.
- [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>>.

8.2. Informative References

- [I-D.dai-mpls-rsvp-te-mbb-label-reuse]
Dai, M. and M. Chaudhry, "MPLS RSVP-TE MBB Label Reuse",
draft-dai-mpls-rsvp-te-mbb-label-reuse-01 (work in
progress), September 2015.
- [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>>.

- [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>>.
- [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>>.
- [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>>.
- [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>>.

Authors' Addresses

Abhishek Deshmukh
Juniper Networks, Inc.
10 Technology Park Dr
Westford, MA 01886
USA

Email: adeshmukh@juniper.net

Kireeti Kompella
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
USA

Email: kireeti@juniper.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 30, 2018

I. Bryskin
Huawei Technologies
X. Liu
Volta Networks
Y. Lee
Huawei Technologies
June 28, 2018

SF Aware TE Topology YANG Model
draft-ietf-teas-sf-aware-topo-model-01

Abstract

This document describes a YANG data model for TE network topologies that are network service and function aware.

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 December 30, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
1.1. Terminology	3
1.2. Tree Diagrams	3
1.3. Prefixes in Data Node Names	3
2. Modeling Considerations	4
3. Model Structure	5
4. YANG Modules	6
5. Model Structure	13
6. YANG Modules	15
7. IANA Considerations	20
8. Security Considerations	21
9. References	21
9.1. Normative References	21
9.2. Informative References	23
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations	24
A.1. SF Aware TE Topology State Module	24
Appendix B. Data Examples	26
B.1. A Topology with Multiple Connected Network Functions	26
B.2. A Topology with an Encapsulated Network Service	31
Authors' Addresses	35

1. Introduction

Today a network offers to its clients far more services than just connectivity across the network. Large variety of physical, logical and/or virtual service functions, network functions and transport functions (collectively named in this document as SFs) could be allocated for and assigned to a client. As described in [I-D.ietf-teas-use-cases-sf-aware-topo-model], there are some important use cases, in which the network needs to represent to the client SFs at the client's disposal as topological elements in relation to other elements of a topology (i.e. nodes, links, link and tunnel termination points) used by the network to describe itself to the client. Not only would such information allow for the client to auto-discover the network's SFs available for the services provisioned for the client, it would also allow for the client selecting the SFs, dual-optimizing the selection on the SF location on the network and connectivity means (e.g. TE tunnels) to inter-connect the SFs. Consequently this would give to both the network and the client powerful means for the service function chain (SFC [RFC7498] [RFC7665]) negotiation to achieve most efficient and cost effective (from the network point of view) and most optimal yet satisfying all necessary constraints of SFCs (from the client's point of view).

This document defines a YANG data model that allows service functions to be represented along with TE topology elements.

1.1. Terminology

The keywords "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].

The following terms are defined in [RFC7950] and are not redefined here:

- o augment
- o data model
- o data node

1.2. Tree Diagrams

A simplified graphical representation of the data model is presented in this document, by using the tree format defined in [I-D.ietf-netmod-yang-tree-diagrams].

1.3. Prefixes in Data Node Names

In this document, names of data nodes, actions, 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
inet	ietf-inet-types	[RFC6991]
nw	ietf-network	[I-D.ietf-i2rs-yang-network-topo]
nt	ietf-network-topology	[I-D.ietf-i2rs-yang-network-topo]
tet	ietf-te-topology	[I-D.ietf-teas-yang-te-topo]

Table 1: Prefixes and Corresponding YANG Modules

2. Modeling Considerations

The model introduced in this document is an augmentation of the TE Topology model defined in [I-D.ietf-teas-yang-te-topo]. SFs are modeled as child elements of a TE node similarly to how Link Termination Points (LTPs) and Tunnel Termination Points (TTPs) are modeled in the TE Topology model. The SFs are defined as opaque objects identified via topology unique service-function-id's. Each SF has one or more Connection Points (CPs) identified via SF-unique sf-connection-point-id's, over which the SF could be connected to other SFs resided on the same TE node, as well as to other elements of the TE node, in particular, to the node's LTPs and/or TTPs. An interested client may use service-function-id's to look up the SFs in TOSCA or YANG data store(s) defined by [ETSI-NFV-MAN] to retrieve the details of the SFs, for example, to understand the SF's mutual substitutability.

The TE Topology model introduces a concept of Connectivity Matrix (CM), and uses the CM to describe which and at what costs a TE node's LTPs could be inter-connected internally across the TE node. The model defined in this document heavily uses the same concept to describe the SF connectivity via introducing 3 additional CMs:

1. SF2SF CM. This CM describes which pairs of SFs could be locally inter-connected, and, if yes, in which direction, via which CPs and at what costs. In other words, the SF2SF CM describes how SFs residing on the same TE node could be inter-connected into local from the TE node's perspective SFCs;
2. SF2LTP CM. This CM describes how, in which direction and at what costs the TE node's SFs could be connected to the TE node's LTPs and hence to SFs residing on neighboring TE nodes that are connected to LTPs at the remote ends of corresponding TE links;
3. SF2TTP CM. This CM describes how, in which direction and at what costs the TE node's SFs could be connected to the TE node's TTPs and hence to SFs residing on other TE nodes on the topology that could be inter-connected with the TE node in question via TE tunnels terminated by the corresponding TTPs.

In addition to SF2SF CM, the local SF chaining could be described with the help of ETSI models Virtual Links (VLs) [ETSI-NFV-MAN]. This option is especially useful when the costs of the local chaining are negligible as compared to ones of the end-to-end SFCs said local SFCs are part of.

Section 3 and 4 provide the YANG model structure and the YANG module for SF-aware Topology. Section 5 and 6 provide the YANG model

structure and the YANG module for Data Center Compute Node resource abstraction. This provides an example of SF2LTP CM where DC compute nodes are connected to LTPs at the remote ends of the corresponding TE links. This use-case is described in Section 12 of [I-D.ietf-teas-use-cases-sf-aware-topo-model].

3. Model Structure

```

module: ietf-te-topology-sf
  augment /nw:networks/nw:network/nw:network-types/tet:te-topology:
    +--rw sf!
  augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes:
      +--rw service-function
        +--rw connectivity-matrices
          +--rw connectivity-matrix* [id]
            +--rw id                uint32
            +--rw from
              | +--rw service-function-id?    string
              | +--rw sf-connection-point-id? string
            +--rw to
              | +--rw service-function-id?    string
              | +--rw sf-connection-point-id? string
            +--rw enabled?          boolean
            +--rw direction?        connectivity-direction
            +--rw virtual-link-id?  string
        +--rw link-terminations
          +--rw link-termination* [id]
            +--rw id                uint32
            +--rw from
              | +--rw tp-ref?    -> ../../../../../../..
            /nt:termination-point/tp-id
              +--rw to
                | +--rw service-function-id?    string
                | +--rw sf-connection-point-id? string
              +--rw enabled?    boolean
              +--rw direction?  connectivity-direction
          augment /nw:networks/nw:network/nw:node/tet:te
            /tet:information-source-entry:
              +--ro service-function
                +--ro connectivity-matrices
                  +--ro connectivity-matrix* [id]
                    +--ro id                uint32
                    +--ro from
                      | +--ro service-function-id?    string
                      | +--ro sf-connection-point-id? string
                    +--ro to
                      | +--ro service-function-id?    string

```

```

    |         |  +--ro sf-connection-point-id?  string
    |         |  +--ro enabled?                 boolean
    |         |  +--ro direction?              connectivity-direction
    |         |  +--ro virtual-link-id?        string
+--ro link-terminations
    +--ro link-termination* [id]
        +--ro id                uint32
        +--ro from
        +--ro to
            |  +--ro service-function-id?      string
            |  +--ro sf-connection-point-id?   string
        +--ro enabled?          boolean
        +--ro direction?        connectivity-direction
augment /nw:networks/nw:network/nw:node/tet:te
/tet:tunnel-termination-point:
    +--rw service-function
    +--rw tunnel-terminations
        +--rw tunnel-termination* [id]
            +--rw id                uint32
            +--rw service-function-id? string
            +--rw sf-connection-point-id? string
            +--rw enabled?          boolean
            +--rw direction?        connectivity-direction

```

4. YANG Modules

```

<CODE BEGINS> file "ietf-te-topology-sf@2018-02-27.yang"
module ietf-te-topology-sf {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-sf";

  prefix "tet-sf";

  import ietf-network {
    prefix "nw";
  }

  import ietf-network-topology {
    prefix "nt";
  }

  import ietf-te-topology {
    prefix "tet";
  }

  organization

```

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

```
contact
```

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

```
Editors:    Igor Bryskin
            <mailto:Igor.Bryskin@huawei.com>
```

```
            Xufeng Liu
            <mailto:Xufeng_Liu@jabil.com>";
```

```
description
```

```
"Network service and function aware aware TE topology model.
```

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

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

```
revision 2018-02-27 {
  description "Initial revision";
  reference "TBD";
}
```

```
/*
 * Typedefs
 */
```

```
typedef connectivity-direction {
  type enumeration {
    enum "to" {
      description
        "The direction is uni-directional, towards the 'to'
        entity direction.";
    }
    enum "from" {
      description
        "The direction is uni-directional, from the 'to'
        entity direction.";
    }
    enum "bidir" {
      description
```

```
        "The direction is bi-directional.";
    }
}
description
    "A type used to indicates whether a connectivity is
    uni-directional, or bi-directional. If the relation is
    uni-directional, the value of this type indicates the
    direction.";
} // connectivity-direction

/*
 * Groupings
 */
grouping service-function-node-augmentation {
    description
        "Augmenting a TE node to be network service and function
        aware.";
    container service-function {
        description
            "Containing attributes related to network services and
            network functions";
        container connectivity-matrices {
            description
                "Connectivity relations between network services/functions
                on a TE node, which can be either abstract or physical.";
            reference
                "ETSI GS NFV-MAN 01: Network Functions Virtualisation
                (NFV); Management and Orchestration.
                RFC7665: Service Function Chaining (SFC) Architecture.";
            list connectivity-matrix {
                key "id";
                description
                    "Represents the connectivity relations between network
                    services/functions on a TE node.";
                leaf id {
                    type uint32;
                    description "Identifies the connectivity-matrix entry.";
                }
            }
        }
        container from {
            description
                "Reference to the source network service or
                network function.";
            leaf service-function-id {
                type string;
                description
                    "Reference to a network service or a network
                    function.";
            }
        }
    }
}
```

```
    }
    leaf sf-connection-point-id {
        type string;
        description
            "Reference to a connection point on a network
            service or a network function.";
    }
} // from
container to {
    description
        "Reference to the destination network service or
        network function.";
    leaf service-function-id {
        type string;
        description
            "Reference to a network service or a network
            function.";
    }
    leaf sf-connection-point-id {
        type string;
        description
            "Reference to a connection point on a network
            service or a network function.";
    }
} // to
leaf enabled {
    type boolean;
    description
        "'true' if this connectivity entry is enabled.";
}
leaf direction {
    type connectivity-direction;
    description
        "Indicates whether this connectivity is
        uni-directional, or bi-directional. If the
        relation is uni-directional, the value of
        this leaf indicates the direction.";
}
leaf virtual-link-id {
    type string;
    description
        "Reference to a virtual link that models this
        connectivity relation in the network function
        model.";
}
} // connectivity-matrix
} // connectivity-matrices
```

```
container link-terminations {
  description
    "Connectivity relations between network services/functions
    and link termination points on a TE node, which can be
    either abstract or physical.";
  reference
    "ETSI GS NFV-MAN 01: Network Functions Virtualisation
    (NFV); Management and Orchestration.
    RFC7665: Service Function Chaining (SFC) Architecture.";
  list link-termination {
    key "id";
    description
      "Each entry of the list represents the connectivity
      relation between a network service/function and
      a link termination point on a TE node.";
    leaf id {
      type uint32;
      description "Identifies the termination entry.";
    }

    container from {
      description
        "Reference to the link termination point.";
    } // from
    container to {
      description
        "Reference to the network service or network
        function.";
      leaf service-function-id {
        type string;
        description
          "Reference to a network service or a network
          function.";
      }
      leaf sf-connection-point-id {
        type string;
        description
          "Reference to a connection point on a network
          service or a network function.";
      }
    } // to
    leaf enabled {
      type boolean;
      description
        "'true' if this connectivity entry is enabled.";
    }
    leaf direction {
      type connectivity-direction;
    }
  }
}
```



```
        description
          "Indicates whether this connectivity is
           uni-directional, or bi-directional. If the
           relation is uni-directional, the value of
           this leaf indicates the direction.";
      }
    } // link-termination
  }
} // service-function-node-augmentation

grouping service-function-ttp-augmentation {
  description
    "Augmenting a tunnel termination point to be network service
     aware.";
  container service-function {
    description
      "Containing attributes related to network services and
       network functions";
    container tunnel-terminations {
      description
        "Connectivity relations between network services/functions
         and tunnel termination points on a TE node, which can be
         either abstract or physical.";
      reference
        "ETSI GS NFV-MAN 01: Network Functions Virtualisation
         (NFV); Management and Orchestration.
         RFC7665: Service Function Chaining (SFC) Architecture.";
      list tunnel-termination {
        key "id";
        description
          "Each entry of the list represents the connectivity
           relation between a network service/function and
           a tunnel termination point on a TE node.";
        leaf id {
          type uint32;
          description "Identifies the termination entry.";
        }

        leaf service-function-id {
          type string;
          description
            "Reference to a network service or a network
             function.";
        }
      }
      leaf sf-connection-point-id {
        type string;
        description

```

```

        "Reference to a connection point on a network
        service or a network function.";
    }
    leaf enabled {
        type boolean;
        description
            "'true' if this connectivity entry is enabled.";
    }
    leaf direction {
        type connectivity-direction;
        description
            "Indicates whether this connectivity is
            uni-directional, or bi-directional. If the
            relation is uni-directional, the value of
            this leaf indicates the direction.";
    }
} // link-termination
}
} // service-function-ttp-augmentation

grouping sf-topology-type {
    description
        "Identifies the SF aware TE topology type.";
    container sf {
        presence "Indidates that the TE topology is SF aware.";
        description
            "Its presence identifies that the TE topology is SF aware.";
    }
} // sf-topology-type

/*
 * Augmentations
 */
/* Augmentations to network-types/te-topology */
augment "/nw:networks/nw:network/nw:network-types/"
+ "tet:te-topology" {
    description
        "Defines the SF aware TE topology type.";
    uses sf-topology-type;
}

/* Augmentations to te-node-attributes */
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes" {
    description
        "Parameters for SF aware TE topology.";
    uses service-function-node-augmentation;
}

```

```

    }

    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:information-source-entry" {
        description
          "Parameters for SF aware TE topology.";
        uses service-function-node-augmentation;
      }

    /* Augmentations to tunnel-termination-point */
    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:tunnel-termination-point" {
        description
          "Parameters for SF aware TE topology.";
        uses service-function-ttp-augmentation;
      }

    /* Augmentations to connectivity-matrix */
    augment "/nw:networks/nw:network/nw:node/tet:te/"
      + "tet:te-node-attributes/tet-sf:service-function/"
      + "tet-sf:link-terminations/tet-sf:link-termination/"
      + "tet-sf:from" {
        description
          "Add reference to the link termination point.
           This portion cannot be shared with the state module.";
        leaf tp-ref {
          type leafref {
            path "../.../nt:termination-point/"
              + "nt:tp-id";
          }
          description
            "Reference to the link termination point.";
        }
      }
    }
  }
}
<CODE ENDS>

```

5. Model Structure

```

module: ietf-cso-dc
  +--rw cso
    +--rw dc* [id]
      |   +--rw hypervisor* [id]
      |   |   +--rw ram
      |   |   |   +--rw total?   uint32
      |   |   |   +--rw used?    uint32

```

```

| | | +--rw free?      uint32
| | +--rw disk
| | | +--rw total?    uint32
| | | +--rw used?     uint32
| | | +--rw free?     uint32
| | +--rw vcpu
| | | +--rw total?    uint16
| | | +--rw used?     uint16
| | | +--rw free?     uint16
| | +--rw instance*  -> /cso/dc/instance/id
| | +--rw id          string
| | +--rw name?       string
+--rw instance* [id]
| | +--rw flavor
| | | +--rw disk?      uint32
| | | +--rw ram?       uint32
| | | +--rw vcpus?    uint16
| | | +--rw id?       string
| | | +--rw name?     string
| | +--rw image
| | | +--rw checksum   string
| | | +--rw size       uint32
| | | +--rw format
| | | | +--rw container? enumeration
| | | | +--rw disk?    enumeration
| | | +--rw id?       string
| | | +--rw name?     string
| | +--rw hypervisor? -> /cso/dc/hypervisor/id
| | +--rw port*       -> /cso/dc/network/subnetwork/port
/id
| | +--rw project?    string
| | +--rw status?     enumeration
| | +--rw id          string
| | +--rw name?       string
+--rw image* [id]
| | +--rw checksum     string
| | +--rw size         uint32
| | +--rw format
| | | +--rw container? enumeration
| | | +--rw disk?      enumeration
| | +--rw id          string
| | +--rw name?       string
+--rw flavor* [id]
| | +--rw disk?        uint32
| | +--rw ram?         uint32
| | +--rw vcpus?      uint16
| | +--rw id          string
| | +--rw name?       string

```

```

|   +--rw dc-monitoring-param* [name]
|   |   +--rw name          string
|   |   +--rw value-string?  string
|   +--rw network* [id]
|   |   +--rw subnetwork* [id]
|   |   |   +--rw port* [id]
|   |   |   |   +--rw ip-address?  inet:ip-address
|   |   |   |   +--rw instance?    -> /cso/dc/instance/id
|   |   |   |   +--rw project?     string
|   |   |   |   +--rw status?      enumeration
|   |   |   |   +--rw id           string
|   |   |   |   +--rw name?        string
|   |   |   +--rw project?  string
|   |   |   +--rw status?    enumeration
|   |   |   +--rw id        string
|   |   |   +--rw name?     string
|   |   +--rw dhcp-agent* [id]
|   |   |   +--rw enabled?  boolean
|   |   |   +--rw pools* [ip-address]
|   |   |   |   +--rw ip-address  inet:ip-address
|   |   |   +--rw project?  string
|   |   |   +--rw status?    enumeration
|   |   |   +--rw id        string
|   |   |   +--rw name?     string
|   |   +--rw project?  string
|   |   +--rw status?    enumeration
|   |   +--rw id        string
|   |   +--rw name?     string
|   |   +--rw cso-ref?   -> /cso/cso-id
|   +--rw ap*           -> /actn-vn:actn/ap
/access-point-list/access-point-id
|   +--rw cso-ref?       -> /cso/cso-id
|   +--rw id             string
|   +--rw name?          string
+--rw cso-id?  string

```

6. YANG Modules

```

<CODE BEGINS> file "ietf-cso-dc@2017-01-16.yang"
module ietf-cso-dc
{
  namespace "urn:ietf:params:xml:ns:yang:ietf-cso-dc";
  prefix "dc";

  import ietf-inet-types {
    prefix "inet";

```

```
}

import ietf-actn-vn {
  prefix "actn-vn";
}

revision 2017-01-16 {
  description
    "Initial revision. This YANG file defines
    the reusable base types for CSO DC description.";
  reference
    "Derived from earlier versions of base YANG files";
}

// Abstract models
grouping resource-element {
  leaf id { type string; }
  leaf name { type string; }
}

grouping resource-instance {
  leaf project { type string; }
  leaf status {
    type enumeration {
      enum active;
      enum inactive;
      enum pending;
    }
  }
  uses resource-element;
}

// Compute models
grouping format {
  leaf container {
    type enumeration {
      enum ami;
      enum ari;
      enum aki;
      enum bare;
      enum ovf;
    }
    default bare;
  }
  leaf disk {
    type enumeration {
      enum ami;
      enum ari;
    }
  }
}
```

```
        enum aki;
        enum vhd;
        enum vmdk;
        enum raw;
        enum qcow2;
        enum vdi;
        enum iso;
    }
    default qcow2;
}
}

grouping image {
    leaf checksum { type string; mandatory true; }
    leaf size { type uint32; units 'Bytes'; mandatory true; }

    container format {
        uses format;
    }

    uses resource-element;
}

grouping flavor {
    leaf disk { type uint32; units 'GB'; default 0; }
    leaf ram { type uint32; units 'MB'; default 0; }
    leaf vcpus { type uint16; default 0; }
    uses resource-element;
}

grouping ram {
    leaf total { type uint32; units 'MB'; }
    leaf used { type uint32; units 'MB'; }
    leaf free { type uint32; units 'MB'; }
}

grouping disk {
    leaf total { type uint32; units 'GB'; }
    leaf used { type uint32; units 'GB'; }
    leaf free { type uint32; units 'GB'; }
}

grouping vcpu {
    leaf total { type uint16; }
    leaf used { type uint16; }
    leaf free { type uint16; }
}
```

```
grouping hypervisor {
    container ram {
        uses ram;
    }

    container disk {
        uses disk;
    }

    container vcpu {
        uses vcpu;
    }

    leaf-list instance {
        type leafref { path '/cso/dc/instance/id'; } }
    uses resource-element;
}

grouping instance {
    container flavor { uses flavor; }
    container image { uses image; }
    leaf hypervisor {
        type leafref { path '/cso/dc/hypervisor/id'; } }
    leaf-list port { type leafref {
        path '/cso/dc/network/subnetwork/port/id'; } }
    uses resource-instance;
}

grouping dc-monitoring-param {
    leaf name {
        description "dc-monitoring-param identifier"; type string; }
    leaf value-string {
        description
            "Current value for a string parameter";
        type string;
    }
}

grouping dc {
    list hypervisor {
        key id;
        uses hypervisor;
    }

    list instance {
        key id;
```



```
    uses instance;
  }

  list image {
    key id;
    uses image;
  }

  list flavor {
    key id;
    uses flavor;
  }

  list dc-monitoring-param {
    key "name";
    uses dc-monitoring-param;
  }

  list network {
    key id;
    uses network;
  }

  leaf-list ap { type leafref {
    path
      '/actn-vn:actn/actn-vn:ap/actn-vn:access-point-list/'
      + 'actn-vn:access-point-id';
  }
  }
  leaf cso-ref { type leafref { path "/cso/cso-id"; } }
  uses resource-element;
}

container cso {
  list dc {
    key id;
    uses dc;
  }

  leaf cso-id { type string; }
}

// Network models
grouping ip-address {
  leaf ip-address { type inet:ip-address; }
}
```

```
    grouping dhcp-agent {
      leaf enabled { type boolean; }
      list pools {
        key ip-address;
        uses ip-address;
      }
      uses resource-instance;
    }

    grouping network {
      list subnetwork {
        key id;
        uses subnetwork;
      }
      list dhcp-agent {
        key id;
        uses dhcp-agent;
      }
      uses resource-instance;
      leaf cso-ref { type leafref { path "/cso/cso-id"; } }
    }

    grouping subnetwork {
      list port {
        key id;
        uses port;
      }
      uses resource-instance;
    }

    grouping port {
      leaf ip-address { type inet:ip-address; }
      leaf instance { type leafref { path '/cso/dc/instance/id'; } }
      uses resource-instance;
    }

  }
<CODE ENDS>
```

7. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

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

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

This document registers the following YANG modules in the YANG Module Names registry [RFC7950]:

```
-----  
name:          ietf-te-topology-sf  
namespace:     urn:ietf:params:xml:ns:yang:ietf-te-topology-packet  
prefix:        tet-sf  
reference:     RFC XXXX  
-----
```

```
-----  
name:          ietf-te-topology-sf-state  
namespace:     urn:ietf:params:xml:ns:yang:ietf-te-topology-packet-state  
prefix:        tet-sf-s  
reference:     RFC XXXX  
-----
```

8. Security Considerations

The configuration, state, action and notification data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241]. The data-model by itself does not create any security implications. The security considerations for the NETCONF protocol are applicable. The NETCONF protocol used for sending the data supports authentication and encryption.

9. References

9.1. 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>>.
- [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>>.
- [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>>.
- [ETSI-NFV-MAN]
ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration", ETSI GS NFV-MAN 001 V1.1.1, December 2014.
- [I-D.ietf-teas-use-cases-sf-aware-topo-model]
Bryskin, I., Liu, X., Guichard, J., Lee, Y., Contreras, L., Ceccarelli, D., and J. Tantsura, "Use Cases for SF Aware Topology Models", draft-ietf-teas-use-cases-sf-aware-topo-model-00 (work in progress), June 2018.
- [I-D.ietf-i2rs-yang-network-topo]
Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo-20 (work in progress), December 2017.
- [I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-16 (work in progress), June 2018.
- [I-D.ietf-netmod-revised-datastores]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-10 (work in progress), January 2018.

9.2. Informative References

- [RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <<https://www.rfc-editor.org/info/rfc7498>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [I-D.ietf-netmod-yang-tree-diagrams] Bjorklund, M. and L. Berger, "YANG Tree Diagrams", draft-ietf-netmod-yang-tree-diagrams-06 (work in progress), February 2018.

Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module `ietf-te-topology-sf` defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [I-D.ietf-netmod-revised-datastores]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module, `ietf-te-topology-sf-state`, is defined as state model, which mirrors the module `ietf-te-topology-sf` 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-sf-state`, is redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion module mirrors that of the cooresponding NMDA model, the YANG tree of the companion module is not depicted separately.

A.1. SF Aware TE Topology State Module

```
<CODE BEGINS> file "ietf-te-topology-sf-state@2018-02-27.yang"
module ietf-te-topology-sf-state {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-sf-state";

  prefix "tet-sf-s";

  import ietf-te-topology-sf {
    prefix "tet-sf";
  }

  import ietf-network-state {
    prefix "nw-s";
  }

  import ietf-network-topology-state {
    prefix "nt-s";
  }

  import ietf-te-topology-state {
    prefix "tet-s";
  }

  organization
    "Traffic Engineering Architecture and Signaling (TEAS)"
}
```

```
    Working Group";

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

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

               Xufeng Liu
               <mailto:Xufeng_Liu@jabil.com>";

description
  "Network service and function aware aware TE topology operational
  state model for non-NMDA compliant implementations.

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

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

revision 2018-02-27 {
  description "Initial revision";
  reference "TBD";
}

/*
 * Augmentations
 */
/* Augmentations to network-types/te-topology */
augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
+ "tet-s:te-topology" {
  description
    "Defines the SF aware TE topology type.";
  uses tet-sf:sf-topology-type;
}

/* Augmentations to connectivity-matrix */
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:te-node-attributes" {
  description
    "Parameters for SF aware TE topology.";
  uses tet-sf:service-function-node-augmentation;
```

```

    }

    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
      + "tet-s:information-source-entry" {
        description
          "Parameters for SF aware TE topology.";
        uses tet-sf:service-function-node-augmentation;
      }

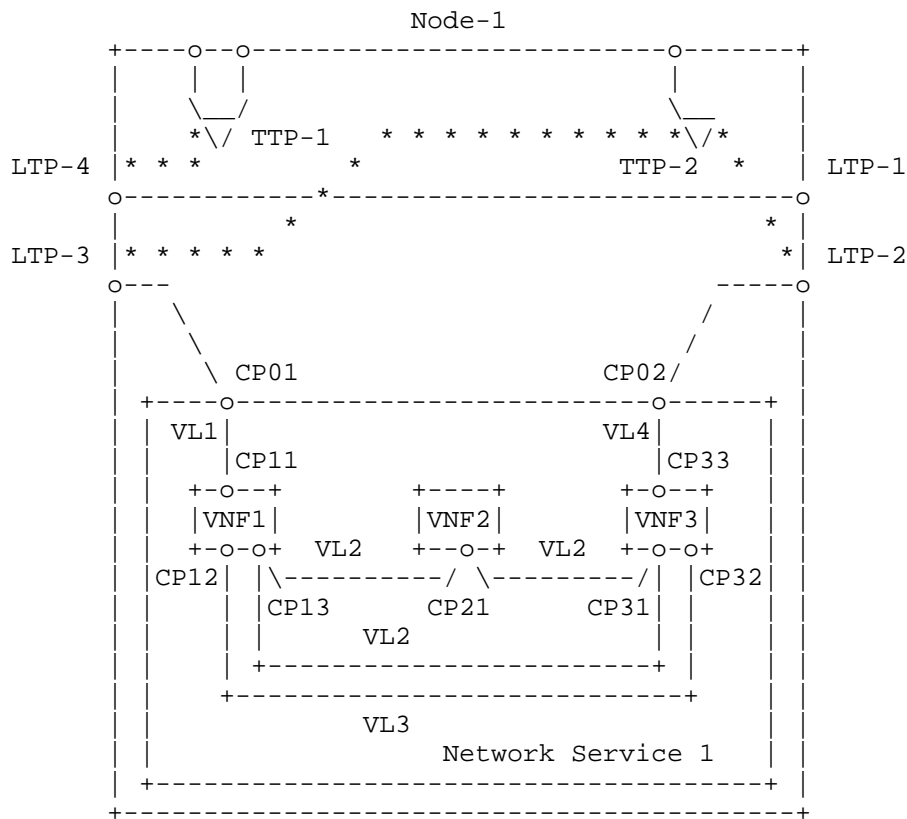
    /* Augmentations to tunnel-termination-point */
    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
      + "tet-s:tunnel-termination-point" {
        description
          "Parameters for SF aware TE topology.";
        uses tet-sf:service-function-ttp-augmentation;
      }

    /* Augmentations to connectivity-matrix */
    augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
      + "tet-s:te-node-attributes/tet-sf-s:service-function/"
      + "tet-sf-s:link-terminations/tet-sf-s:link-termination/"
      + "tet-sf-s:from" {
        description
          "Add reference to the link termination point.
           This portion cannot be shared with the state module.";
        leaf tp-ref {
          type leafref {
            path "../.../.../.../nt-s:termination-point/"
              + "nt-s:tp-id";
          }
        }
        description
          "Reference to the link termination point.";
      }
    }
  }
}
<CODE ENDS>

```

Appendix B. Data Examples

B.1. A Topology with Multiple Connected Network Functions



The configuration instance data for Node-1 in the above figure could be as follows:

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {
            "sf": {}
          }
        },
        "network-id": "network-sf-aware",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:network-sf-aware",
        "node": [
          {
            "node-id": "Node-1",
```

```
"te-node-id": "2.0.1.1",
"te": {
  "te-node-attributes": {
    "domain-id": 1,
    "is-abstract": [null],
    "connectivity-matrices": {
    },
    "service-function": {
      "connectivity-matrices": {
        "connectivity-matrix": [
          {
            "id": 10,
            "from": {
              "service-function-id": "Network Service 1",
              "sf-connection-point-id": "CP01"
            },
            "to": {
              "service-function-id": "VNF1",
              "sf-connection-point-id": "CP11"
            }
          },
          {
            "id": 13,
            "from": {
              "service-function-id": "VNF1",
              "sf-connection-point-id": "CP12"
            },
            "to": {
              "service-function-id": "VNF3",
              "sf-connection-point-id": "CP32"
            }
          },
          {
            "id": 12,
            "from": {
              "service-function-id": "VNF1",
              "sf-connection-point-id": "CP13"
            },
            "to": {
              "service-function-id": "VNF2",
              "sf-connection-point-id": "CP21"
            }
          }
        ]
      },
      "direction": "bidir",
      "virtual-link-id": "VL1"
    }
  },
  "direction": "bidir",
  "virtual-link-id": "VL1"
},
{
  "id": 13,
  "from": {
    "service-function-id": "VNF1",
    "sf-connection-point-id": "CP12"
  },
  "to": {
    "service-function-id": "VNF3",
    "sf-connection-point-id": "CP32"
  },
  "direction": "bidir",
  "virtual-link-id": "VL3"
},
{
  "id": 12,
  "from": {
    "service-function-id": "VNF1",
    "sf-connection-point-id": "CP13"
  },
  "to": {
    "service-function-id": "VNF2",
    "sf-connection-point-id": "CP21"
  },
  "direction": "bidir",
  "virtual-link-id": "VL2"
}
```

```

    },
    {
      "id": 23,
      "from": {
        "service-function-id": "VNF2",
        "sf-connection-point-id": "CP21"
      },
      "to": {
        "service-function-id": "VNF3",
        "sf-connection-point-id": "CP31"
      },
      "direction": "bidir",
      "virtual-link-id": "VL2"
    },
    {
      "id": 30,
      "from": {
        "service-function-id": "Network Service 1",
        "sf-connection-point-id": "CP02"
      },
      "to": {
        "service-function-id": "VNF3",
        "sf-connection-point-id": "CP33"
      },
      "direction": "bidir",
      "virtual-link-id": "VL4"
    }
  ]
},
"link-terminations": {
  "link-termination": [
    {
      "id": 2,
      "from": {
        "tp-ref": "LTP-2"
      },
      "to": {
        "service-function-id": "Network Service 1",
        "sf-connection-point-id": "CP02"
      },
      "direction": "bidir"
    },
    {
      "id": 3,
      "from": {
        "tp-ref": "LTP-3"
      },
      "to": {

```

```

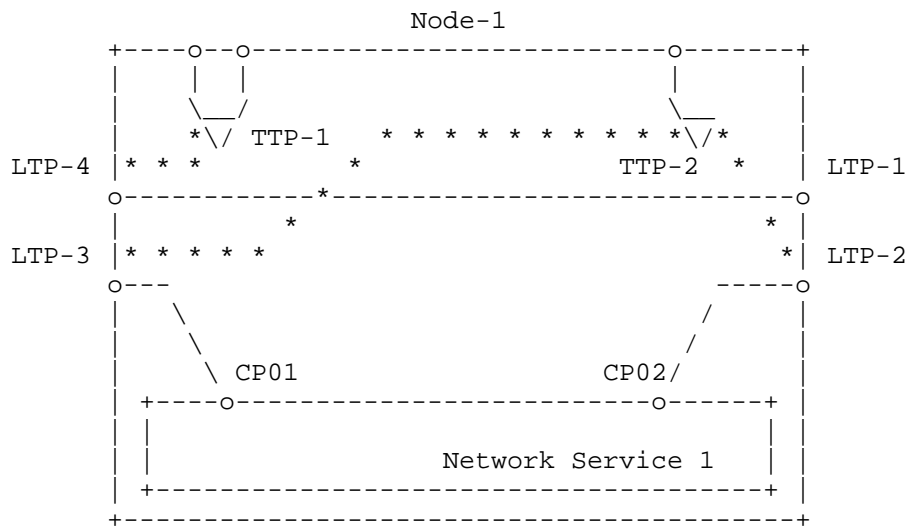
        "service-function-id": "Network Service 1",
        "sf-connection-point-id": "CP01"
    }
    "direction": "bidir"
}
]
}
}
}
}
"tunnel-termination-point": [
{
    "tunnel-tp-id": 10001,
    "name": "TTP-1",
    "service-function-terminations": {
    },
},
{
    "tunnel-tp-id": 10002,
    "name": "TTP-2",
    "service-function-terminations": {
    }
}
]
},
"termination-point": [
{
    "tp-id": "LTP-1",
    "te-tp-id": 10001
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-l2sc",
                "encoding": "lsp-encoding-ethernet"
            }
        ]
    }
},
{
    "tp-id": "LTP-2",
    "te-tp-id": 10002
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-l2sc",
                "encoding": "lsp-encoding-ethernet"
            }
        ]
    }
}
]
}

```

```
{
    "tp-id": "LTP-3",
    "te-tp-id": 10003
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-l2sc",
                "encoding": "lsp-encoding-ethernet"
            }
        ]
    }
},
{
    "tp-id": "LTP-4",
    "te-tp-id": 10004
    "te": {
        "interface-switching-capability": [
            {
                "switching-capability": "switching-l2sc",
                "encoding": "lsp-encoding-ethernet"
            }
        ]
    }
}
]
```

B.2. A Topology with an Encapsulated Network Service

In this example, a network service consists of several interconnected network functions (NFs), and is represented by this model as an encapsulated opaque object without the details between its internals.



The configuration instance data for Node-1 in the above figure could be as follows:

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {
            "sf": {}
          }
        },
        "network-id": "network-sf-aware",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:network-sf-aware",
        "node": [
          {
            "node-id": "Node-1",
            "te-node-id": "2.0.1.1",
            "te": {
              "te-node-attributes": {
                "domain-id": 1,
                "is-abstract": [null],
                "connectivity-matrices": {
                },
              },
              "service-function": {
                "connectivity-matrices": {
                },
              },
            }
          }
        ]
      }
    ]
  }
}
```

```
"link-terminations": {
  "link-termination": [
    {
      "id": 2,
      "from": {
        "tp-ref": "LTP-2"
      },
      "to": {
        "service-function-id": "Network Service 1",
        "sf-connection-point-id": "CP02"
      },
      "direction": "bidir"
    },
    {
      "id": 3,
      "from": {
        "tp-ref": "LTP-3"
      },
      "to": {
        "service-function-id": "Network Service 1",
        "sf-connection-point-id": "CP01"
      },
      "direction": "bidir"
    }
  ]
}

"tunnel-termination-point": [
  {
    "tunnel-tp-id": 10001,
    "name": "TTP-1",
    "service-function-terminations": {
    }
  },
  {
    "tunnel-tp-id": 10002,
    "name": "TTP-2",
    "service-function-terminations": {
    }
  }
]

"termination-point": [
  {
    "tp-id": "LTP-1",
    "te-tp-id": 10001
    "te": {

```

```

        "interface-switching-capability": [
            {
                "switching-capability": "switching-l2sc",
                "encoding": "lsp-encoding-ethernet"
            }
        ]
    },
    {
        "tp-id": "LTP-2",
        "te-tp-id": 10002
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-l2sc",
                    "encoding": "lsp-encoding-ethernet"
                }
            ]
        }
    },
    {
        "tp-id": "LTP-3",
        "te-tp-id": 10003
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-l2sc",
                    "encoding": "lsp-encoding-ethernet"
                }
            ]
        }
    },
    {
        "tp-id": "LTP-4",
        "te-tp-id": 10004
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-l2sc",
                    "encoding": "lsp-encoding-ethernet"
                }
            ]
        }
    }
]
}

```



```
    ]  
  }  
}
```

Authors' Addresses

Igor Bryskin
Huawei Technologies

EMail: Igor.Bryskin@huawei.com

Xufeng Liu
Volta Networks

EMail: xufeng.liu.ietf@gmail.com

Young Lee
Huawei Technologies

EMail: leeyoung@huawei.com

TEAS Working Group
Internet Draft
Intended status: Informational

Igor Bryskin
Huawei Technologies
Vishnu Pavan Beeram
Juniper Networks
Tarek Saad
Cisco Systems Inc
Xufeng Liu
Volta Networks

Expires: January 2, 2019

July 2, 2018

TE Topology and Tunnel Modeling for Transport Networks
draft-ietf-teas-te-topo-and-tunnel-modeling-02

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on January 2, 2019.

Copyright Notice

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

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

Abstract

This document describes how to model TE topologies and tunnels for transport networks, by using the TE topology YANG model [I-D.ietf-teas-yang-te-topo] and the TE tunnel YANG model [I-D.ietf-teas-yang-tel].

Conventions used in this document

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

Table of Contents

1. Modeling Considerations.....	3
1.1. TE Topology Model.....	3
1.2. TE Topology Modeling Constructs.....	5
1.3. Abstract TE Topology Calculation, Configuration and Maintenance.....	22
1.3.1. Single-Node Abstract TE Topology.....	23
1.3.2. Full Mesh Link Abstract TE Topology.....	25
1.3.3. Star-n-Spokes Abstract TE Topology.....	27
1.3.4. Arbitrary Abstract TE Topology.....	28
1.3.5. Customized Abstract TE Topologies.....	29
1.3.6. Hierarchical Abstract TE Topologies.....	30
1.4. Merging TE Topologies Provided By Multiple Providers.....	31
1.4.1. Dealing With Multiple Abstract TE Topologies Provided By The Same Provider.....	34
1.5. Configuring Abstract TE Topologies.....	36
1.6. TE Tunnel Model.....	37
1.7. TE Tunnel/Transport Service Modeling Constructs.....	39
1.8. Transport Service Mapping.....	53
1.9. Multi-Domain Transport Service Coordination.....	54
2. Use Cases.....	59

2.1. Use Case 1. Transport service control on a single layer multi-domain transport network.....	59
2.2. Use Case 2. End-to-end TE tunnel control on a single layer multi-domain transport network.....	67
2.3. Use Case 3. Transport service control on a ODUk/Och multi-domain transport network with Ethernet access links.....	71
2.4. Use Case 4. Transport service control on a ODUk/Och multi-domain transport network with multi-function access links.....	78
2.5. Use Case 5. Real time updates of IP/MPLS layer TE link attributes that depend on supporting transport connectivity (e.g. transport SRLGs, propagation delay, etc.).....	81
2.6. Use Case 6. Virtual Network Service.....	82
3. Security Considerations.....	85
4. IANA Considerations.....	85
5. References.....	86
5.1. Normative References.....	86
5.2. Informative References.....	86
6. Acknowledgments.....	86
Appendix A. Data Examples.....	87
A.1. Use Case 1.....	87
A.1.1. Domain 1.....	87
A.1.2. Domain 2.....	94
A.1.3. Domain 3.....	100
Authors' Addresses.....	106

1. Modeling Considerations

1.1. TE Topology Model

The TE Topology Model is written in YANG modeling language. It is defined and developed by the IETF TEAS WG and is documented as "YANG Data Model for TE Topologies" [I-D.ietf-teas-yang-te-topo]. The model describes a TE network provider's Traffic Engineering data store as it is seen by a client. It allows for the provider to convey to each of its clients:

- o information on network resources available to the client in the form of one or several native TE topologies (for example, one for each layer network supported by the provider);
- o one or several abstract TE topologies, customized on per-client basis and sorted according to the provider's preference as to how the abstract TE topologies are to be used by the client;
- o updates with incremental changes happened to the previously provided abstract/native TE topology elements;

- o updates on telemetry/state information the client has expressed interest in;
- o overlay/underlay relationships between the TE topologies provided to the client (e.g. TE path computed in an underlay TE topology supporting a TE link in an overlay TE topology);
- o client/server inter-layer adaptation relationships between the TE topologies provided to the client in the form of TE inter-layer locks or transitional links;

The TE Topology Model allows a network client to:

- o (Re-)configure/negotiate abstract TE topologies provided to the client by a TE network provider, so that said abstract TE topologies optimally satisfy the client's needs, constraints and optimization criteria, based on the client's network planning, service forecasts, telemetry information extracted from the network, previous history of service provisioning and performance monitoring, etc.;
- o Obtain abstract/native TE topologies from multiple providers and lock them horizontally (inter-domain) and vertically (inter-layer) into the client's own native TE topologies;
- o Configure, with each provider the trigger, frequency and contents of the TE topology update notifications;
- o Configure, with each provider the trigger, frequency and contents of the TE topology telemetry (e.g. statistics counters) update notifications.

1.2. TE Topology Modeling Constructs

Figure 1. TE Topology

- o TE domain - a multi-layer traffic engineered network under direct and complete control of a single authority, network provider. TE domain can be described by one or more TE topologies. For example, separate TE topologies can describe each of the domain's layer networks. TE domain can hierarchically encompass/parent other (child) TE domains, and can be encompassed by its own parent.
- o TE topology - a graphical representation of a TE domain. TE topology is comprised of TE nodes (TE graph vertices) interconnected via TE links (TE graph edges).

```

/* TE topology */
augment /nw:networks/nw:network:
  /* TE topology global ID */
  +--rw provider-id?      te-types:te-global-id
  +--rw client-id?        te-types:te-global-id
  +--rw te-topology-id?   te-types:te-topology-id
  .....
  /* TE topology general parameters */
  |   +--rw preference?          uint8
  |   +--rw optimization-criterion? identityref
  .....

```

```
        /* TE topology list of TE nodes */
augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?   te-types:te-node-id
.....
        /* TE topology list of TE links */
augment /nw:networks/nw:network/nt:link:
.....
        /* TE topology list of TE link termination points */
augment /nw:networks/nw:network/nw:node/nt:termination-point:
  +--rw te-tp-id?   te-types:te-tp-id
.....
```

Figure 2. TE Node

- o TE node - an element of a TE topology (appears as a vertex on TE graph). A TE node represents one or several nodes (physical switches), or a fraction of a node. A TE node belongs to and is fully defined in exactly one TE topology. A TE node is assigned a TE topology scope-unique ID. TE node attributes include information related to the data plane aspects of the associated node(s) (e.g. TE node's connectivity matrix), as well as configuration data (such as TE node name). A given TE node can be reached on the TE graph, representing the TE topology, over one of TE links terminated by the TE node.

```

/* TE node */
augment /nw:networks/nw:network/nw:node:
  /* TE node ID */
  +--rw te-node-id?    te-types:te-node-id
  .....
  /* TE node general attributes */
  |   +--rw te-node-attributes */
  .....
  /* TE node connectivity matrices */
  |   +--rw connectivity-matrices
  .....
  /* TE node underlay TE topology */
  |   +--rw underlay-topology {te-topology-hierarchy}?
  |   +--rw network-ref?    leafref
  .....
  /* TE node information sources*/
  |   +--ro information-source-entry* [information-source]
  .....
  /* TE node statistics */
  +--ro statistics
  .....
  /* TE node TTP list */
  +--rw tunnel-termination-point* [tunnel-tp-id]
  .....

```

- o TE link - an element of a TE topology (appears as an edge on TE graph), TE link is unidirectional and its arrow indicates the TE link's direction. Edges with two arrows on the TE topology graph (see Figure 1) represent bi-directional combinations of two parallel oppositely directed TE links. A TE link represents one or several physical links or a fraction of a physical link. A TE link belongs to and is fully defined in exactly one TE topology. A TE link is assigned a TE topology scope-unique ID. 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.) A TE link is connected to a TE node, terminating the TE link via exactly one TE link termination point (LTP).

```

/* TE link */
augment /nw:networks/nw:network/nt:link:
/* TE link bundle information */
    |   |--rw (bundle-stack-level)?
    |   |   |--rw bundled-links
    |   |   |--rw component-links
    .....
/* TE link general attributes */
    |   |--rw te-link-attributes
    .....
/* TE link underlay TE topology */
    |   |--rw underlay! {te-topology-hierarchy}?
    |   |   |--rw primary-path
    |   |   |--rw backup-path* [index]
    .....
/* TE link layer network */
    |   |--rw interface-switching-capability* [switching-
capability encoding]
    .....
/* TE link protection type */
    |   |   |--rw protection-type?   uint16
    .....

/* TE link supporting TE tunnels */
    |   |   |--rw tunnels

```

```

.....
/* TE link transitional link flag */
|   +--ro is-transitional?           empty

.....
/* TE link information sources */
|   +--ro information-source?       te-info-source

.....
/* TE link statistics */
|   +--ro statistics

.....

```

- o Intra-domain TE link - TE link connecting two TE nodes within the same TE topology representing a TE network domain (e.g. L14 in Figure 1). From the point of view of the TE topology where the intra-domain TE link is defined, the TE link is close-ended, that is, both local and remote TE nodes of the link are defined in the same TE topology.
- o Inter-domain TE link - TE link connecting two border TE nodes that belong to separate TE topologies describing neighboring TE network domains (e.g. L3x in Figure 1). From the point of view of the TE topology where the inter-domain TE link is defined, the TE link is open-ended, that is, the remote TE node of the link is not defined in the TE topology where the local TE node and the TE link itself are defined.

[Note: from the point of view of a TE node terminating an inter-domain TE link there is no difference between inter-domain and access TE links]

- o Access TE link - TE link connecting a border TE node of a TE topology describing a TE network domain to a TE node of a TE topology describing a customer network site (e.g. L1x in Figure 1). From the point of view of the TE topology where the access TE link is defined, the TE link is open-ended, that is, the remote TE node of the link (t.e. TE node representing customer network element(s)) is not defined in the TE topology where the local TE node and the TE link itself are defined.

[Note: from the point of view of a TE node terminating an access TE link there is no difference between access and inter-domain TE links]

- o Dynamic TE link - a TE link that shows up in (and disappears from) a TE topology as a result of multi-layer traffic engineering. Dynamic TE link (supported by a hierarchy TE tunnel dynamically set up in a server layer network) is automatically (i.e. without explicit configuration request) added to a client layer network TE topology to augment the topology with additional flexibility to ensure successful completion of the path computation for and provisioning of a client layer network connection/LSP. For example, an ODUk hierarchy TE tunnel can support a dynamic Ethernet layer TE link to enable provisioning of an Ethernet layer connection on a network that does not have sufficient static Ethernet layer connectivity. Likewise, dynamic TE link is automatically removed from the TE topology (and its supporting hierarchy TE tunnel released) as soon as the TE link stops carrying client layer connections/LSPs.
- o TE link termination point (LTP) - a conceptual point of connection of a TE node to one of the TE links terminated by the TE node (see Figure 2a). Unlike TE link, LTP is bi-directional - an inbound TE link and an oppositely directed outbound TE link have to be connected to the TE node via the same LTP to constitute a bi-directional TE link combination.

Figure 2a. Bi-directional TE link combination (left), independent uni-directional TE links (right)

```

/* LTP */
augment /nw:networks/nw:network/nw:node/nt:termination-point:
/* LTP ID */
  +--rw te-tp-id?    te-types:te-tp-id
/* LTP network layer ID */
  | +--rw interface-switching-capability* [switching-
capability encoding]
  | | +--rw switching-capability    identityref

```

```

| | +--rw encoding                               identityref
/* LTP bandwidth information */
| | +--rw max-lsp-bandwidth* [priority]
| |   +--rw priority      uint8
| |   +--rw bandwidth?    te-bandwidth
/* LTP inter-layer locks */
| +--rw inter-layer-lock-id?                      uint32
.....

```

- o TE tunnel termination point (TTP) - an element of TE topology representing one or several potential TE tunnel termination/adaptation points (e.g. OCh layer transponder). A TTP is hosted by exactly one TE node (see Figure 2). A TTP is assigned a TE node scope-unique ID. 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 originated/terminated from/by the TE node. TTP's important attributes include Local Link Connectivity List, Adaptation Client Layer List, TE inter-layer locks (see below), Unreserved Adaptation Bandwidth (announcing the TTP's remaining adaptation resources sharable between all potential client LTPs), and Property Flags (indicating miscellaneous properties of the TTP, such as capability to support 1+1 protection for a TE tunnel terminated on the TTP).

```

/* TTP */
+--rw tunnel-termination-point* [tunnel-tp-id]
/* TTP ID */
+--rw tunnel-tp-id                               binary
/* TTP layer network ID */
| +--rw switching-capability?                     identityref
| +--rw encoding?                                 identityref
/* Inter-layer-locks supported by TTP */
| +--rw inter-layer-lock-id?                      uint32
/* TTP's protection capabilities */
| +--rw protection-type?                          identityref
/* TTP's list of client layer users */
| +--rw client-layer-adaptation
.....
/* TTP's Local Link Connectivity List (LLCL) */

```

```
|  +-rw local-link-connectivities
.....

```

o Label - in the context of circuit switched layer networks identifies a particular resource on a TE link (e.g. Och wavelength, ODUk container)

```
+-:-(label)
  +-rw value?   rt-types:generalized-label
```

Figure 3. TTP Local Link Connectivity List

- o TTP basic local link connectivity list (basic LLCL) - a list of TE link/label combinations terminated by the TTP-hosting TE node (effectively the same as LTP/label pairs), which the TTP could be connected to (see Figure 3, upper left). From the point of view of a potential TE path, basic LLCL provides a list of permissible LTP/label pairs the TE path needs to start/stop on for a connection, taking the TE path, to be successfully terminated on the TTP in question.
- o TTP detailed local link connectivity list (detailed LLCL) - basic LLCL extended to provide a set of costs (such as intra-node summary TE metric, delay, SRLGs, etc.) associated with each LLCL entry (see Figure 3, upper right)

```

/* TTP LLCL */
|  +--rw local-link-connectivities
|  |      +--rw number-of-entries?          uint16
|  /* LLCL entry */
|
|  /* LLCL entry LTP */
|  |      +--rw link-tp-ref                  leafref
|
.....

/* LLC entry label range */
|  +--rw label-restriction* [inclusive-exclusive label-start]
|  |      +--rw inclusive-exclusive          enumeration
|  |      +--rw label-start                  rt-types:generalized-label
|  |      |      +--rw label-end?            rt-types:generalized-
label
|  |      |      |      +--rw range-bitmap?    binary
|  |      |
|
/* LLCL entry underlay TE path(s) */
|  +--rw underlay! {te-topology-hierarchy}?
|  |      +--rw primary-path
|  |      +--rw backup-path* [index]
/* LLCL entry protection type */
|  |      +--rw protection-type?          uint16
/* LLCL entry supporting TE tunnels */
|  |      +--rw tunnels
/* LLCL entry bandwidth parameters */
|  |      +--rw max-lsp-bandwidth* [priority]
|
.....

```

```

/* LLCL entry metrics (vector of costs) */
|   +--rw te-default-metric?          uint32
|   +--rw te-delay-metric?            uint32
|   +--rw te-srlgs
|   |   +--rw value*      te-types:srlg
|   +--rw te-nsrlgs {nsrlg}?

```

```

.....
/* LLCL entry ID */
|   |   +--rw id*      uint32

```

- o TTP adaptation client layer list - a list of client layers that could be directly adopted by the TTP. This list is necessary to describe complex multi-layer (more than two layer) client-server layer hierarchies and, in particular, to identify the position of the TTP in said hierarchies.

```

/* TTP adaptation client layer list */
|   +--rw client-layer-adaptation
|   |   +--rw switching-capability* [switching-capability
encoding]
/* Client layer ID */
|   |   +--rw switching-capability      identityref
|   |   +--rw encoding                  identityref
/* Adaptation bandwidth available for the client layer */
|   |   +--rw bandwidth?                te-bandwidth

```

Figure 4. TE Node Connectivity Matrix

- o TE node basic connectivity matrix - a TE node attribute describing the TE node's switching capabilities/limitations in the form of permissible switching combinations of the TE node's LTP/label pairs (see Figure 4, upper left). From the point of view of a potential TE path arriving at the TE node at a given inbound LTP/label, the node's basic connectivity matrix describes permissible outbound LTP/label pairs for the TE path to leave the TE node.
- o TE node detailed connectivity matrix - TE node basic connectivity matrix extended to provide a set of costs (such as intra-node summary TE metric, delay, SRLGs, etc.) associated with each connectivity matrix entry (see Figure 4, upper right).

```

/* TE node connectivity matrix */
    |  +--rw connectivity-matrix* [id]
    |  +--rw id                               uint32

```



```

|      +--rw from /* left LTP */
|      |      +--rw tp-ref?   leafref
|      +--rw to /* right LTP */
|      |      +--rw tp-ref?   leafref
|      +--rw is-allowed?      boolean

/* Connectivity matrix entry label range */
|      +--rw label-restriction* [inclusive-exclusive
label-start]
|      |      +--rw inclusive-exclusive   enumeration
|      |      +--rw label-start          rt-
types:generalized-label
|      |      +--rw label-end?           rt-
types:generalized-label
|      |      +--rw range-bitmap?        binary

/* Connectivity matrix entry underlay TE path(s) */
|      +--rw underlay! {te-topology-hierarchy}?
|      |      +--rw primary-path
|      |      +--rw backup-path* [index]
/* Connectivity matrix entry protection type */
|      |      +--rw protection-type?   uint16
/* Connectivity matrix entry supporting TE tunnels */
|      |      +--rw tunnels
/* Connectivity matrix entry bandwidth parameters */
|      +--rw max-lsp-bandwidth* [priority]

.....
/* Connectivity matrix entry metrics (vector of costs) */
|      +--rw te-default-metric?        uint32
|      +--rw te-delay-metric?          uint32
|      +--rw te-srlgs
|      |      +--rw value*   te-types:srlg
|      +--rw te-nsrlgs {nsrlg}?

.....
/* Connectivity matrix entry ID */
|      |      +--rw id*   uint32

```

Figure 5. TE Path

- o TE path - an ordered list of TE node/link IDs (each possibly augmented with labels) that interconnects over a TE topology a pair of TTPs and could be used by a connection (see Figure 5). A TE path could, for example, be a product of a successful path computation performed for a given TE tunnel

```

/* TE path */

/* TE topology the path is defined in */
| | | +--rw network-ref?    leafref
/* Path type (IRO, XRO, ERO, RRO) */
| | | +--rw path-type?     identityref

/* TE path elements */
| | | +--rw path-element* [path-element-id]
| | | |   +--rw path-element-id    uint32
| | | |   +--rw index?              uint32
| | | |   +--rw (type)?
/* Numbered TE link path element */
| | | |   +--:(ip-address)
| | | | |   +--rw ip-address-hop
| | | | |   +--rw address?         inet:ip-address

```

```
| | | +---rw hop-type?      te-hop-type  
| | | /* AS number path element */  
| | |     +---:(as-number)  
| | |         +---rw as-number-hop  
| | |             +---rw as-number?      binary  
| | |                 +---rw hop-type?    te-hop-type  
| | | /* Unnumbered TE link path element */  
| | |     +---:(unnumbered-link)  
| | |         +---rw unnumbered-hop  
| | |             +---rw te-node-id?      inet:ip-address  
| | |                 +---rw tp-id?       uint32  
| | |                     +---rw hop-type?    te-hop-type  
| | | /* Label path element */  
| | |     +---:(label)  
| | |         +---rw label-hop  
| | |             +---rw value?          rt-types:generalized-label  
| | |                 +---rw direction?   boolean  
| | |     +---:(sid)  
| | |         +---rw sid-hop  
| | |             +---rw sid?            rt-types:generalized-label
```

- o TE path segment - a contiguous fragment of a TE path

Figure 6. TE Inter-Layer Lock

- o TE inter-layer lock - a modeling concept describing client-server layer adaptation relationships important for 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. A 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 by a given TE inter-layer lock share the same inter-layer lock ID value.

In Figure 6 a TE inter-layer lock IL_1 associates six client layer LTPs (C_LTP_1 - C_LTP_6) with two server layer TTPs (S_TTP_1 and S_TTP_2). As mentioned, they all have the same attribute -inter-layer lock ID: IL_1, which is the only parameter/value indicating the association. A given LTP may have zero, one or more inter-layer lock IDs. In the case of multiple inter-layer lock IDs, this implies that the data arriving at the LTP can be adopted onto any of TTPs associated with all specified inter-layer locks. For example, C_LTP_1 may be attributed with two inter-layer locks- IL_1 and IL_2. This would mean that C_LTP_1 for adaptation purposes can use not just TTPs associated with inter-layer lock IL_1 (i.e. S_TTP_1 and S_TTP_2 in the Figure), but any of TTPs associated with inter-layer lock IL_2. Likewise, a given TTP may have one or more inter-layer locks, meaning that it can offer the adaptation service to any client layer LTP having an inter-layer lock matching one of its own.

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

```
|  +--rw inter-layer-lock-id?          uint32
```

- o Transitional link - an alternative method of modeling of client-server adaptation relationship. Transitional link is a bi-directional link connecting an LTP in a client layer to an LTP in a server layer, which is associated (via TTP's LLCL) with a server layer TTP capable of adopting of the client layer data onto a TE tunnel terminated by the TTP. Important attributes of a transitional link are local/remote LTP IDs, TE metric and available adaptation bandwidth.

Figure 7. Native and Abstract TE Topologies

- o Native TE topology - a TE topology as it is known (to full extent and unmodified) to the TE topology provider (see lower part of Figure 7.). A native TE topology might be discovered via various routing protocols and/or subscribe/publish techniques. For example, a first-level TE topology provider (such as a T-SDN Domain Controller, DC) may auto-discover its native TE topology(ies) by participating in the domain's OSPF-TE protocol instance; while a second-level TE topology provider (such as a Hierarchical T-SDN Controller, HC) normally builds its native TE topology(ies) based on TE topologies exposed by each of the subordinate, first-level TE topology providers.
- o Underlay TE topology - a TE topology that serves as a base for constructing overlay TE topologies.

- o Overlay TE topology - a TE topology constructed based on one or more underlay TE topologies. Each TE node of the overlay TE topology represents a separate underlay TE topology (that could be mapped onto an arbitrary segment of a native TE topology). Each TE link of the overlay TE topology represents, generally speaking, an arbitrary TE path in one of the underlay TE topologies. The overlay TE topology and the supporting underlay TE topologies may represent separate layer networks (e.g. OTN/ODUk and WDM/OCh respectively) or the same layer network.
- o Abstract TE topology - an overlay TE topology created by a provider to describe its network in some abstract way. An abstract TE topology contains at least one abstract TE topology element, such as TE node or TE link. An abstract TE topology is built based on contents of one or more of the provider's native TE topologies (serving as underlay(s)), the provider's policies and the client's preferences (see upper part of Figure 7).
- o Customized TE topology - a TE topology tailored for a given provider's client. A customized TE topology is usually but not always an abstract TE topology. For example, a given abstract TE topology could be exposed to a group or all provider's clients (in which case the abstract TE topology is not a customized TE topology). Likewise, a given naive TE topology could be customized for a given client (for example, by removing high delay TE links the client does not care about). So customized TE topology is not an abstract TE topology, because it does not contain abstract TE topology elements
- o TE inter-domain plug - a TE link attribute meaningful for open-ended inter-domain/access TE links. It contains a network-wide unique value (inter-domain plug ID) that identifies in the network a connectivity supporting the inter-domain/access TE link in question. It is expected that a given pair of neighboring domain TE topologies (provided by separate providers) will have each at least one open-ended inter-domain/access TE link with a TE inter-domain plug matching to one provided by its neighbor, thus allowing for a client of both domains to identify adjacent nodes in the separate neighboring TE topologies and resolve the open-ended inter-domain/access TE links by connecting them regardless of the links respective local/remote node ID/link ID attributes. 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).

```

+--rw external-domain
|   +--rw network-ref?          leafref
|   +--rw remote-te-node-id?    te-types:te-node-id
|   +--rw remote-te-link-tp-id? te-types:te-tp-id
|   +--rw plug-id?              uint32

```

1.3. Abstract TE Topology Calculation, Configuration and Maintenance

The TE Topology Model does not prescribe what and how abstract TE topologies are computed, configured, manipulated and supported by a TE network (e.g. transport network) provider. However, it is assumed that:

- o All TE topologies, native or abstract, conveyed to the same or different clients, are largely independent one from another. This implies that each TE topology, generally speaking, has an independent name space for TE node and link IDs, SRLGs, etc. (possibly overlapping with the name spaces of other TE topologies);
- o All abstract TE topologies are bound to the respective underlay native or abstract TE topologies only by the overlay/underlay relationships defined by the TE Topology Model, but, otherwise, the abstract TE topologies are decoupled from their respective underlay TE topologies.

It is envisioned that an original set of abstract TE topologies is produced by a TE network provider for each of its clients based on the provider's local configurations and/or policies, as well as the client-specific profiles. The original set of abstract TE topologies offered to a client may be accepted by the client as-is. Alternatively, the client may choose to negotiate/re-configure the abstract TE topologies, so that the latter optimally satisfy the client's needs. In particular, for each of the abstract TE topologies the client may request adding/removing TE nodes, TE links, TTPs and/or modifying re-configurable parameters of the existing components. The client may also request different optimization criteria as compared to those used for the original abstract TE topology optimization, or/and specify various topology-level constraints. The provider may accept or reject all or some abstract TE topology re-configuration requests. Hence, the abstract TE topology negotiation process may take multiple iterations before the provider and each of its clients agree upon a set of abstract TE

topologies and their contents. Furthermore, the negotiation process could be repeated over time to produce new abstract TE topologies optimal to best suit evolving circumstances.

Figure 8. Native Transport Network Domain TE Topology as an Underlay for Abstract TE Topologies

Let's assume that a native transport network domain TE topology to be as depicted in Figure 8. The popular types of abstract TE topologies based on this native TE topology as an underlay are described in the following sections.

1.3.1. Single-Node Abstract TE Topology

Figure 9. Blocking/Asymmetrical TE Node with Basic Connectivity Matrix Attribute

In Figure 9, the transport network domain is presented to a client as a one-node abstract TE topology, where the single TE node (AN1) represents the entire domain and terminates all of the inter-domain/access TE links connecting the domain to its adjacent domains (i.e. TE links L1...L8). Because AN1 represents the entire domain the node's Underlay TE Topology attribute matches the ID of one of the domain's native TE topologies (e.g. one presented in Figure 8). [Note: all or some of the underlay TE topologies a given abstract TE topology depends on could be catered to the client by the provider along with the abstract TE topology in question or upon separate request(s) issued by the client.]

One important caveat about abstract TE node AN1 is that it should be considered as an asymmetrical/blocking switch, because, generally speaking, it is not guaranteed that a suitable TE path exists between any given pair of inter-domain TE links into/out of the domain. This means from the TE Topology model point of view that there are certain limitations as to how AN1's LTPs could be interconnected inside/across the TE node. The model allows for asymmetrical/blocking switches by specifying for the associated TE nodes a non-empty basic connectivity matrix attribute describing permissible inbound-outbound TE link/label switching combinations. It is assumed that the provider's path computer can compute a set of optimal TE paths, connecting inbound TE link/label_x <=> outbound TE link/label_y combinations inside the abstract TE node over the TE node's underlay TE topology. Based on the results of such computations, AN1's connectivity matrix can be (re-)generated and (re-)conveyed to the abstract TE topology client.

A richer version of the basic connectivity matrix is the detailed connectivity matrix. The latter not only describes permissible inbound TE link/label_x <=> TE link/label TE link/label_y switching combinations, but also provides connectivity matrix entry specific vectors of various costs/metrics (in terms of delay, bandwidth, intra-node SRLGs and summary TE metrics) that a potential TE path will accrue, should a given connectivity matrix entry be selected by the path for crossing the TE node (see Figure 10).

Figure 10. Blocking/Asymmetrical TE Node with Detailed Connectivity
Matrix Attribute

1.3.2. Full Mesh Link Abstract TE Topology

Figure 11. Full Mesh Link Abstract TE Topology

In Figure 11, the transport network domain is abstracted in the following way.

- o Each of the underlay native TE topology border TE nodes (i.e., the TE nodes terminating at least one inter-domain/access TE link, such as TE nodes S3 or S11 in Figure 8) is represented in the abstract TE topology as a separate abstract TE node, matching one-for-one to the respective border TE node of the underlay TE topology. For example, S3' of the abstract TE topology represents S3 of the underlay TE topology in Figure 8. [Note that such a relationship is modeled via Supporting Node attribute of TE node S3' specifying the ID of S3, as well as the ID of the TE topology where S3 is defined (i.e. TE topology in Figure 8)]. Likewise, S9' represents S9, S11' represents S11 and so forth;
- o TE nodes S3', S5', S8', S9' and S11' are interconnected via a full mesh of abstract TE links. It is assumed that the provider's path computer can compute a set of optimal TE paths over one or more of underlay TE topologies (such as presented in Figure 8)- one for each of said abstract TE links; and the provider can set up the TE tunnels in the network supporting each of the abstract TE links, either during the abstract TE topology configuration (in the case of committed/pre-established abstract TE links), or at the time the first client's connection is placed on the abstract TE link in question (the case of uncommitted abstract TE links). [Note that so (re-)computed TE paths, as well as the IDs of respective underlay TE topologies used for their computation are normally catered to the client in the Underlay TE path attribute of the associated abstract TE links]

The configuration parameters of each of the abstract TE links (such as layer ID, bandwidth and protection requirements, preferred TE paths across the underlay TE topology for the primary and backup connections, etc.) are expected to be found in the abstract TE topology profiles/templates locally configured with the provider or pushed to the provider by the client via the policy NBI. Each of the abstract TE links may be later re-configured or removed by direct configuration requests issued by the client via TE Topology NBI. Likewise, additional abstract TE links may be requested by the client at any time.

Some possible variants/flavors of the Full Mesh Link Abstract TE Topology described above are:

- o Partial Mesh Link Abstract TE Topology (where some of the abstract TE links from the full mesh are missing);
- o Double Mesh Link Abstract TE Topology (where each pair of abstract TE nodes is connected via two diverse abstract TE links).

1.3.3. Star-n-Spokes Abstract TE Topology

Figure 12. Star-n-Spoke Abstract TE Topology

The Full Mesh Link Abstract TE Topology suffers from the n -squared problem; that is, the number of required abstract TE links is proportional to square of the number of native TE topology border TE nodes. This problem can be mitigated (i.e., the number of required abstract TE links may be significantly reduced) by adding, to the abstract TE topology, an additional abstract TE node (the star) representing one or several interconnected non-border TE nodes from the native TE topology. Abstract TE links in the Star-n-Spokes Topology connect the star with all other TE nodes of the topology (the spokes). For example, abstract TE node AN1 in Figure 12 could represent collectively TE nodes S7, S10 and S4 of the native TE topology (see Figure 8) with abstract TE links connecting AN1 with all other TE nodes in the Star-n-Spokes Abstract TE Topology in Figure 12.

In order to introduce a composite abstract TE node, (e.g. AN1 in Figure 12) representing in a given abstract TE topology an arbitrary segment of another TE topology (e.g. TE nodes S7, S12 and S4 of the TE topology in Figure 8) the TE topology provider is expected to perform the following operations:

- o Copy the TE topology segment to be represented by the abstract TE node (i.e. TE nodes S7, S10 and S4 in Figure 8, as well as the TE links interconnecting them) into a separate auxiliary TE topology (with a separate TE topology ID);

- o Set for each TE node and TE link of the auxiliary TE topology the Supporting Node/Link attribute matching the original TE topology ID, as well as the ID of the respective original TE node/link of the original TE topology. For example, if S7" of the auxiliary TE topology is a copy of S7 of the original TE topology, the Supporting Node attribute of S7" will specify the ID of the original TE topology (presented in figure 8) and the ID of S7;
- o Set for the abstract TE node AN1 the Underlay TE Topology attribute matching the auxiliary TE Topology ID

Furthermore, the Star-n-Spokes Abstract TE topology provider is expected to:

- o Compute/provision TE paths/tunnels supporting each of the abstract TE links in Figure 12 (i.e. abstract TE links connecting the spokes to the star, AN1) as described in 1.3.2;
- o Generate the AN1's Basic/Detailed Connectivity Matrix attribute based on intra-node path computations performed on the AN1's underlay (i.e. auxiliary) TE topology and describing permissible inbound TE link/label_x. outbound TE link/label_y switching combinations as described in 1.3.1

1.3.4. Arbitrary Abstract TE Topology

Figure 13. Arbitrary Abstract TE Topology

To achieve an optimal tradeoff between the number of components, the amount of information exposed by a transport network provider and the

amount of path computations required to keep said information up-to-date, the provider may present the TE network domain as an arbitrary abstract TE topology comprised of any number of abstract TE nodes interconnected by abstract TE links (see Figure 13). Each of the abstract TE nodes can represent a single or several interconnected TE nodes from the domain's underlay (native or lower level abstract) TE topology, or a fraction of an underlay TE node. [Note that each of the abstract TE nodes of the TE topology in Figure 13 is expected to be introduced and maintained by the provider following the instructions as described in 1.3.3; likewise, each of the abstract TE links of the topology is expected to be computed, provisioned and maintained as described in 1.3.2]

1.3.5. Customized Abstract TE Topologies

Figure 14. Customized Abstract TE Topology(ies)

A transport network/domain provider may serve more than one client. In such a case, the provider "slices" the network/domain resources and exposes a slice for each of the clients in the form of a customized abstract TE topology. In Figure 14, the provider serves

two clients (Blue and Red). Client Blue is provided with the Blue abstract TE topology supported by the blue TE tunnels or paths in the underlay (native) TE topology (depicted in the Figure with blue broken lines). Likewise, client Red is provided with the Red abstract TE topology supported by the red TE tunnels or paths in the underlay TE topology.

1.3.6. Hierarchical Abstract TE Topologies

Figure 15. Hierarchy of Abstract TE Topologies

As previously mentioned, an underlay TE topology for a given abstract TE topology component does not have to be one of the domain's native TE topologies - another (lower level) domain's abstract TTE topology can be used instead. This means that abstract TE topologies are hierarchical in nature.

Figure 15 provides an example of abstract TE topology hierarchy. In this Figure the blue topology is a top level abstract TE topology catered to by the provider to one of the domain's clients. One of the TE links of the blue topology - link EF - is supported by a TE path E'-M-P-Q-N-F' computed in the underlay TE topology (red topology), which happens to be domain's (lower level) abstract TE topology.. Furthermore, as shown, the TE link PQ - one of the TE links comprising the E'-M-P-Q-N-F' path - is supported by its own underlay

TE path, P'-X-Q' - computed on one of the domain's native TE topologies.

Importantly, each TE link and TE node of a given abstract TE topology has, generally speaking, its individual stack/hierarchy of underlay TE topologies.

1.4. 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 an interconnected multi-domain transport network. In order to make use of said topologies, the client is expected to merge (inter-connect) the provided TE topologies into one or more client's native TE topologies, each of which homogeneously representing the multi-domain transport network. This makes it possible for the client to select end-to-end TE paths for its TE tunnel connections traversing multiple domains.

In particular, the process of merging TE topologies includes:

- o Identifying neighboring TE domains and locking their TE topologies horizontally by connecting their inter-domain open-ended TE links;
- o Renaming TE node, link, and SRLG IDs into 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. Original TE node/link IDs along with the original TE topology ID are stored in the Source attribute of the respective TE nodes/links of the merged TE topology;
- o Locking, TE topologies associated with different layer networks vertically according to provided TE inter-layer locks; this is to facilitate inter-layer path computations across multiple TE topologies provided by the same topology provider.

Figure 16. Merging Domain TE Topologies

Figure 16 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 a TE topology (abstract or native), describing the network domain under the respective provider's control. The client, by consulting the attributes of the open-ended inter-domain/access TE links - such as TE inter-domain plugs or remote TE node/link IDs - is able to determine that:

1. the two domains are adjacent and are interconnected via three inter-domain TE links, and;
2. 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 interconnects the open-ended TE links, as shown on the upper part of the Figure.

As mentioned, one way to interconnect the open-ended inter-domain/access TE links of neighboring domains is to mandate the providers to specify remote nodeID/linkID attributes in the provided inter-domain/access TE links. This, however, may prove to be not flexible. For example, the providers may not be aware of the respective remote nodeID/linkID values. More importantly, this option does not allow for the client to mix-n-match multiple (more than one) TE topologies catered by the same providers (see the next section). Another, more flexible, option to resolve the open-ended inter-domain/access TE links is by decorating them with the TE inter-domain plug attribute. The attribute specifies inter-domain plug ID - a network-wide unique value that identifies on the network connectivity supporting a given inter-domain/access TE link. Instead of specifying remote node ID/link ID, an inter-domain/access 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/access TE link with a TE inter-domain plug matching to one provided by its neighbor. For example, the inter-domain TE link originating from node S5 of the Domain 1 TE topology (Figure 8) and the inter-domain TE link coming from node S3 of Domain2 TE topology may specify matching TE inter-domain plugs (i.e. carrying the same inter-domain plug ID). This would allow for the client to identify adjacent nodes in the separate neighboring TE topologies and resolve the inter-domain/access 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. IDs of the original (i.e. abstract TE topology) TE nodes/links along with the ID of the abstract TE topology they belong to are stored in the Source attribute of the respective TE nodes/links of the merged TE topology. For example, the Source attribute of S27 will contain S7 and the TE topology ID of the abstract TE topology describing domain 2.

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.

1.4.1. Dealing With Multiple Abstract TE Topologies Provided By The Same Provider

Figure 17. Multiple Abstract TE Topologies Provided By TE Topology Providers

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 connections. 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 to decide how to mix-and-match multiple abstract TE topologies provided by each 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 on the upper part of Figure 16, the client may merge the abstract TE topologies received from the two providers, as shown in Figure 17, into the client's additional native TE topologies, as shown in Figure 18.

[Note: allowing for the client mix-n-matching of multiple TE topologies assumes that TE inter-domain plugs (rather than remote nodeID/linked) option is used for identifying neighboring domains and inter-domain/access TE link resolution.]

Figure 18. Multiple Native (Merged) Client's TE Topologies

It is important to keep in mind 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 connections. The choice as to which topology to use for a given connection depends on the connection/tunnel parameters/requirements and the topology's style and optimization criteria.

1.5. Configuring Abstract TE Topologies

When a client receives one or more abstract TE topologies from one of its providers, it may accept the topologies as-is and merge them into one or more of its own native TE topologies. Alternatively, the client may choose to request a re-configuration of one, some or all abstract TE topologies provided by the providers. Specifically, with respect to a given abstract TE topology, some of its TE nodes/links may be requested to be removed, while additional ones may be requested to be added. It is also possible that existing TE nodes/links may be asked to be re-configured. For example, a set of TE links may be requested to be disjoint from each other by configuring the same Non Sharing Risk Link Group (NSRLG) attribute for all links from the set. Such a configuration would force the provider to place TE tunnels supporting the TE links from the set onto sufficiently disjoint TE paths computed in the tunnels underlay TE topology. Furthermore, the topology-wide optimization criteria may be requested to be changed. For example, underlay TE paths supporting the abstract TE links, currently optimized to be shortest (least-cost) paths, may be requested to be re-optimized based on the minimal delay criteria. Additionally, the client may request the providers to configure entirely new abstract TE topologies and/or to remove existing ones. Furthermore, future periodic or one time additions, removals and/or re-configurations of abstract TE topology elements and/or their attributes could be (re-)scheduled by the client ahead of time.

It is the responsibility of the client to implement the logic behind the above-described abstract TE topology negotiation. It is expected that the logic is influenced by the client's local configuration/templates, policies conveyed by client's clients, input from the network planning process, telemetry processor, analytics systems and/or direct human operator commands. Figure 19 exemplifies the abstract TE topology negotiation process. As shown in the Figure, the original abstract TE topology exposed by a provider was requested to be re-configured. Specifically, one of the abstract TE links was asked to be removed, while three new ones were asked to be added to the abstract TE topology.

Figure 19. Provider. Client Abstract TE Topology Negotiation

1.6. TE Tunnel Model

The TE Tunnel Model is written in YANG modeling language. It is defined and developed by the IETF TEAS WG and is documented as "YANG Data Model for Traffic Engineering Tunnels and Interfaces" [I-D.ietf-teas-yang-te]. Among other things the model describes a TE network provider's TE Tunnel data store as it is seen and influenced by a client.

The TE Tunnel Model allows for the provider to convey to each of its clients:

- o information on TE tunnels provided to the client that are fully contained within the controlled network domain,
- o information on multi-domain TE tunnel segments across the network domain controlled by the provider;
- o information on connections/LSPs, supporting TE tunnels and TE tunnel segments;
- o updates in response to changes to the client's active TE tunnels/segments and the connections supporting them,

- o updates in response to the TE tunnel/segment telemetry/state information the client has expressed an interest in.

The TE Tunnel Model allows for a TE network client to:

- o Issue configuration requests to set up, tear down, replace, modify and manipulate end-to-end TE tunnels, as well as segments of multi-domain TE tunnels across the network controlled by the provider;
- o Request and obtain information on active TE tunnels/segments and connections supporting them;
- o Subscribe to and configure with the provider triggers, pace and contents of the TE tunnel/segment change update notifications;
- o Subscribe to and configure with the provider triggers, pace and contents of the TE tunnel/segment event notifications, such as detected alarms, faults, protection/restoration actions, etc..
- o Subscribe to and configure with the provider triggers, pace and contents of TE tunnel/segment telemetry (e.g. statistics counters) update notifications.

1.7. TE Tunnel/Transport Service Modeling Constructs

Figure 20. TE tunnel

- o TE tunnel - a connection-oriented service provided by a layer network of delivery of a client's data between source and destination tunnel termination points. A TE tunnel in a server layer network may support a link in a client layer network (e.g. OCh layer TE tunnel supporting ODU4 link). In Figure 20, a TE tunnel interconnects tunnel termination points resident on switches C-R2 and C-R3. A TE tunnel is realized via (supported by, mapped onto) one or more layer network connections/LSPs

```
/* TE tunnel */
|  +--rw tunnel* [name]
|  |  +--rw name                               leafref
```



```

| | | +--ro oper-status? identityref
/* TE tunnel primary path and LSP container */
| | | +--rw p2p-primary-paths
| | | +--rw p2p-primary-path* [name]
| | | +--rw name
| | | /* Configuration */
leafref
| | | +--rw config
| | | +--rw name? string
| | | +--rw preference? uint8
| | | +--rw path-setup-protocol? identityref
| | | +--rw path-computation-method? identityref
| | | +--rw path-computation-server? inet:ip-
address
| | | +--rw compute-only? empty
| | | +--rw use-cspf? boolean
| | | +--rw verbatim? empty
| | | +--rw lockdown? empty
| | | +--rw named-explicit-path? leafref
| | | +--rw named-path-constraint? leafref {te-
types:named-path-constraints}?
| | | /* state */
| | | +--ro state
| | | +--ro name? string
| | | +--ro preference? uint8
| | | +--ro path-setup-protocol? identityref
| | | +--ro path-computation-method? identityref
| | | +--ro path-computation-server? inet:ip-
address
| | | +--ro compute-only? empty
| | | +--ro use-cspf? boolean
| | | +--ro verbatim? empty
| | | +--ro lockdown? empty
| | | +--ro named-explicit-path? leafref
| | | +--ro named-path-constraint? leafref
{te-types:named-path-constraints}?
| | | /* Computed path */
| | | /* Computed path properties/metrics /
| | | +--ro computed-path-properties
| | | | +--ro path-metric* [metric-type]
| | | | | +--ro metric-type identityref
| | | | | +--ro accumulative-value? uint64
| | | /* Computed path affinities */
| | | +--ro path-affinities
| | | | +--ro constraints* [usage]
| | | | | +--ro usage?
identityref

```

						+--ro (style)?			
						+---:(value)			
						+--ro value?		te-	
types:admin-groups									
						+---:(named)			
[name]						+--ro affinity-names*			
						+--ro name	string		
						/* Computed path SRLGs */			
						+--ro path-srlgs			
						+--ro (style)?			
						+---:(values)			
						+--ro usage?	identityref		
types:srlg						+--ro values*	te-		
						+---:(named)			
						+--ro constraints* [usage]			
identityref						+--ro usage			
						+--ro constraint			
						+--ro srlg-names* [name]			
						+--ro name	string		
						/* Computed path sub-objects */			
						+--ro path-computed-route-objects			
								
						/* LSP (provisioned path) */			
						+--ro lsp* [source destination tunnel-id			
lsp-id extended-tunnel-id type]									
						/* LSP parameters */			
						+--ro source	leafref		
						+--ro destination	leafref		
						+--ro tunnel-id	leafref		
						+--ro lsp-id	leafref		
						+--ro extended-tunnel-id	leafref		
						+--ro type	leafref		
						+--ro signaling-type?	identityref		
						+--rw candidate-p2p-secondary-paths			
						+--rw candidate-p2p-secondary-path*			
[secondary-path]									
						+--rw secondary-path	leafref		
						+--rw config			
						+--rw secondary-path?	leafref		
						+--rw priority?	uint16		
identityref						+--rw path-setup-protocol?			
						+--ro state			
						+--ro secondary-path?	leafref		

- o Tunnel termination point (TTP) - a physical device inside a given node/switch realizing a TE tunnel termination function in a given layer network, as well as the TE tunnel's adaptation function provided for client layer network(s). One example of tunnel termination point is an OCh layer transponder. [Note: Tunnel termination points are not to be confused with TE tunnel termination points, which are TE representations of physical tunnel termination points. Similar to physical switches and links of the network, such as depicted in Figure 20, being represented on a TE topology describing the network as TE nodes and TE links, (physical) tunnel termination points (TTPs) are represented as TE tunnel termination points (TE TTPs, see 1.2) hosted by the TE nodes. For example, a provisioned connection/LSP starts on a source TTP, goes through a chain of physical links and stops on a destination TTP. In contrast, TE path (e.g. result of a path computation) starts on a source TE TTP, goes through a chain of TE links and stops on a destination TE TTP.]

			---rw source?	inet:ip-address
			---rw destination?	inet:ip-address
			---rw src-tp-id?	binary
			---rw dst-tp-id?	binary

- o TE tunnel hand-off point - an access link or inter-domain link by which a multi-domain TE tunnel enters or exits a given network domain, in conjunction with a layer network resource (such as a wavelength channel or ODUk container) allocated on the access/inter-domain link for the TE tunnel.
- o TE tunnel segment - a part of a multi-domain TE tunnel that spans a given network domain and is directly and fully controlled by the domain's controller, DC. TE tunnel segment is a fragment of a multi-domain TE tunnel between
 1. the source tunnel termination point and the TE tunnel hand-off point outbound from the TE tunnel's first domain (head TE tunnel segment);
 2. inbound and outbound TE tunnel hand-off points into/from a given domain (transit TE tunnel segment);

3. inbound TE tunnel hand-off point into the TE tunnel's last domain and the destination tunnel termination point (tail TE tunnel segment);
- o Transport service - the same as TE tunnel segment
 - o Hierarchy TE tunnel - a server layer TE tunnel that supports a dynamically created TE link in the client layer network topology (e.g. see 1.2)

```

/* Hierarchy TE tunnel parameters */
      | | | | |
      | | | | | +--rw hierarchical-link-id
      | | | | | +--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-id?         te-types:te-
topology-id

```

- o Hierarchy transport service - the first or the last segment of a multi-domain hierarchy TE tunnel
- o Dependency TE tunnel - a hierarchical TE tunnel provisioned or to be provisioned in an immediately adjacent server layer a given client layer TE tunnel depends on (i.e. carried or to be carried within)
- o Potential TE tunnel/segment - a TE tunnel/segment configured in COMPUTE_ONLY mode. For such a TE tunnel/segment TE paths to be taken by supporting connection(s) is/are computed and monitored, but the connection(s) are not provisioned

```

      | | | | | +--rw path-computation-method?  identityref
      | | | | | +--rw path-computation-server?  inet:ip-
address
      | | | | | +--rw compute-only?             empty
      | | | | | +--rw use-cspf?                 Boolean

```

Figure 20a. TE Tunnel Connections/LSPs

- o Layer network connection/connection/LSP - a layer network path supporting a TE tunnel by realizing its implied forwarding function. Said path is provisioned in a given layer network's data plane over a chain of links and cross-connected over switches terminating the links. It interconnects the supported TE tunnel's source and destination termination points (in the case of end-to-end connection) or TE tunnel's hand-off points (in the case of transport service connection) or the TE tunnel's two split-merge points (in the case of segment protection connection).

Example: ODU2 connection supporting an ODU2 TE tunnel.

				/* LSP (provisioned path) */	
				+--ro lsp* [source destination tunnel-id	
lsp-id	extended-tunnel-id	type]			
				/* LSP parameters */	
				+--ro source	leafref
				+--ro destination	leafref
				+--ro tunnel-id	leafref
				+--ro lsp-id	leafref
				+--ro extended-tunnel-id	leafref
				+--ro type	leafref
				+--ro signaling-type?	identityref
.....				
				+--ro priority?	uint16
				+--ro path-setup-protocol?	
identityref					
				+--ro active?	Boolean

- o Working connection - the primary connection of the supported TE tunnel or transport service (see Figure 20a).
- o End-to-end protection connection - a secondary end-to-end connection of the supported TE tunnel (e.g. end-to-end 1+1 protection connection, see Figure 20a).
- o Segment protection connection - a secondary connection of the supported transport service protecting the service over a given network domain (e.g. 1+1 segment protection connection, see Figure 20a)
- o Restored connection - a connection after successful network failure restoration procedures
- o Current connection - the same as restored connection
- o Nominal connection - a connection as (re-)provisioned upon a client configuration request (i.e. a connection before any automatic network failure restoration re-configurations are carried out, also a connection after restoration reversion procedures are successfully completed)
- o Unprotected TE tunnel/transport service - TE tunnel/transport service supported by a single (working/primary) connection/LSP

- o Protected TE tunnel/transport service - TE tunnel/transport service supported by one working connection/LSP and at least one protection/secondary connection/LSP
- o Restorable TE tunnel/transport service - TE tunnel/transport service with pre-configured automatic network failure restoration capabilities
- o TE tunnel/transport service automatic protection switchover - a process of switching of carrying user payload from the tunnel's/service's affected by a network failure working connection onto one of the tunnel's/service's healthy protection connection
- o TE tunnel/transport service automatic protection reversion - a process of switching of carrying user payload from the tunnel's/service's protection connection back onto the tunnel's/service's working connection after the latter was repaired from network failure
- o TE tunnel/transport service protection external command - a command, typically issued by an operator, which influences the automatic protection switchover and reversion.

External commands are defined in [ITU-T G.800] and [RFC 4427]:

- . Freeze: A temporary configuration action that prevents any switch action to be taken and as such freezes the current state.
- . Clear Freeze: An action that clears the active Freeze state.
- . Lockout of Normal: A temporary configuration action that ensures that the normal traffic is not allowed to use the protection transport entity.

As described in [ITU-T G.808], this command should be issued at both ends.

- . Clear Lockout of Normal: An action that clears the active Lockout of Normal state.
- . Lockout of Protection: A temporary configuration action that ensures that the protection transport entity is temporarily not available to transport a traffic signal (either normal or extra traffic).

- . Forced Switch: A switch action that swithes the extra traffic signal, the normal traffic signal, or the null signal to the protection transport entity, unless an equal or higher priority switch command is in effect.
 - . Manual Switch: A switch action that switches the extra traffic signal, the normal traffic signal #i, or the null signal to the protection transport entity, unless a fault condition exists on other transport entities or an equal or higher priority switch command is in effect.
 - . Exercise: An action to start testing if the APS communication is operating correctly. It is lower priority than any other state or command.
 - . Clear: An action that clears the active near-end lockout of protection, forced switch, manual switch, WTR state, or exercise command
- o TE tunnel/transport service protection Hold-off time - a configured period of time to expire between the moment of detecting of the first network failure affecting the tunnel's/service's working connection and the begining of the tunnel's/service's automatic protection switchover procedures
 - o TE tunnel/transport service protection WTR time - a configured period of time to expire between the moment of repairing the last network failure affecting the tunnel's/service's working connection and the begining of the tunnel's/service's automatic protection reversion procedures
 - o TE tunnel/transport service automatic network failure restoration - a process of replacing of the tunnel's/service's connection(s) affected by one or more network failures away from the point(s) of failue
 - o TE tunnel/transport service restoration reversion- a process of replacing of the tunnel's/service's connection(s) back onto the nominal connection paths after all network failures affecting the tunnel's/service's nominal connection(s) are repaired
 - o TE tunnel/transport service restoration Hold-off time - a configured period of time to expire between the moment of detecting of the first network failure affecting the tunnel's/service's nominal or current connection and the beginning of the automatic connection restoration procedures

- o TE tunnel/transport service restoration WTR time - a configured period of time to expire between the moment of repairing the last network failure affecting the tunnel's/service's nominal connection and the beginning of the connection automatic restoration reversion procedures
- o Configured restoration path - a TE path specified by the client to be used during the automatic network failure restoration operation on one of the TE tunnel's/transport service's nominal or current connections
- o Pre-computed restoration path - a configured restoration path to be validated by a path computer during the TE tunnel/transport service setup or client triggered modification
- o Pre-provisioned restoration path - a pre-computed restoration path to be pre-provisioned/pre-signaled in the network (with all associated network resources allocated but not necessarily bound into cross-connects) during the TE tunnel/transport service setup or client triggered modification
- o Connection configured path - a TE path (see 1.2) over a TE topology describing a layer network/domain that specifies (loosely or strictly) the client's requirements with respect to an ordered list of network nodes, links and resources on the links a given connection should go through

			+++rw explicit-route-object* [index]	
			+++rw index	leafref
			+++rw explicit-route-usage?	identityref
(INCLUDE/EXCLUDE)				
			+++rw index?	uint32
			+++rw (type)?	
			+++:(numbered)	
				+++rw numbered-hop
				+++rw address?
id				te-types:te-tp-
				+++rw hop-type?
				te-hop-type
			+++:(as-number)	
				+++rw as-number-hop
				+++rw as-number?
				binary
				+++rw hop-type?
				te-hop-type
			+++:(unnumbered)	
				+++rw unnumbered-hop

node-id					+++rw node-id?	te-types:te-
tp-id					+++rw link-tp-id?	te-types:te-
					+++rw hop-type?	te-hop-type
					+++:(label)	
					+++rw label-hop	
types:generalized-label					+++rw value?	rt-
					+++:(sid)	
					+++rw sid-hop	
types:generalized-label					+++rw sid?	rt-

- o Connection exclusion path - a TE path over a TE topology describing a layer network/domain that specifies the client's requirements with respect to an unordered list of network nodes, links and resources on the links to be avoided by a given connection

					+++rw route-object-exclude-always* [index]	
					+++rw index	leafref
					+++rw index?	uint32
					+++rw (type)?	
					+++:(numbered)	
					+++rw numbered-hop	
id					+++rw address?	te-types:te-tp-
					+++:(as-number)	
					+++rw as-number-hop	
					+++rw as-number?	binary
					+++:(unnumbered)	
					+++rw unnumbered-hop	
node-id					+++rw node-id?	te-types:te-
tp-id					+++rw link-tp-id?	te-types:te-
					+++:(label)	
					+++rw label-hop	
types:generalized-label					+++rw value?	rt-

```

      |           |   |   |
types:generalized-label  +--:(sid)
                        +--rw sid-hop
                        +--rw sid?   rt-

```

- o Connection computed path - a TE path over a TE topology describing a layer network/domain as computed (subject to all configured constraints and optimization criteria) for a given connection to take. Computed connection path could be thought as the TE path intended to be taken by the connection

```

/* Computed path */
/* Computed path properties/metrics /
    | | |
    | | | +---ro computed-path-properties
    | | |     +---ro path-metric* [metric-type]
    | | |         +---ro metric-type          identityref
    | | |         +---ro accumulative-value?   uint64
/* Computed path affinities */
    | | | +---ro path-affinities
    | | |     +---ro constraints* [usage]
    | | |         +---ro usage?
identityref
    | | |         +---ro (style)?
    | | |             +---:(value)
    | | |                 | +---ro value?          te-
types:admin-groups
    | | |             +---:(named)
    | | |                 +---ro affinity-names*
[name]
    | | |             +---ro name          string
/* Computed path SRLGs */
    | | | +---ro path-srlgs
    | | |     +---ro (style)?
    | | |         +---:(values)
    | | |             +---ro usage?          identityref
    | | |             +---ro values*        te-
types:srlg
    | | |         +---:(named)
    | | |             +---ro constraints* [usage]
identityref
    | | |                 +---ro usage

```

```

| | | | | +--ro constraint
| | | | | +--ro srlg-names* [name]
| | | | | +--ro name      string
| | | | | /* Computed path sub-objects */
| | | | | +--ro path-computed-route-objects
.....

```

- o Connection actual path - an active connection's path as provisioned in the layer network's data plane in the form of a TE path over a TE topology describing the layer network/domain

1.8. Transport Service Mapping

Figure 21. Transport Service Mapping

Let's assume that a provider has exposed to a client its network domain in the form of an abstract TE topology, as shown on the left side of Figure 21. From then on, the provider should be prepared to receive from the client, a request to set up or manipulate a transport service with TE path(s) computed for the service connection(s) based on and expressed in terms of the provided abstract TE topology (as, for example, displayed in red broken line on the right side of Figure 21). When this happens, the provider is expected to set up the TE tunnels supporting all yet uncommitted abstract TE links (e. g, TE link S3'-S8' in the Figure).

Furthermore, it is the responsibility of the provider to:

- o Perform all the necessary abstract-to-native translations for the specified TE paths (i.e. the transport service connection configured paths);

- o Provision working and protection connections supporting the transport service; as well as replace/modify/delete them in accordance with subsequent client's configuration requests;
- o Perform all the requested recovery operations upon detecting network failures affecting the transport service;
- o Notify the client about all parameter changes, events and other telemetry information the client has expressed an interest in, with respect to the transport service in question.

1.9. Multi-Domain Transport Service Coordination

A client of multiple TE network domains may need to orchestrate/coordinate its transport service setup/manipulation across some or all the domains. One example of such a client is a Hierarchical T-SDN Controller, HC, managing a connected multi-domain transport network where each of the domains is controlled by a separate Domain T-SDN Controller, DC. Said DCs are expected to expose TE Topology and TE Tunnel North Bound Interfaces, NBIs,, supported respectively by IETF TE Topology and TE Tunnel models (and their network layer specific augmentations). HC is assumed to establish client-provider relationship with each of the DCs and make use of said NBIs to extract from the domains various information (such as TE topologies and telemetry), as well as to convey instructions to coordinate across multiple domains its transport services set up and manipulation.

Figure 22. Two-Domain Transport Network

Let's consider, for example, a two-domain transport network as represented in Figure 22. Suppose that HC is requested to set up an unprotected transport service to provide connectivity between customer network elements C-R1 and C-R6. It is assumed that by the time the request has arrived, the two DCs have already provided abstract TE topologies describing their respective domains, and that HC has merged the provided TE topologies into one that homogeneously describes the entire transport network (as shown in Figure 23).

Figure 23. Two-Domain Transport Network (Abstracted View)

Consider that HC, using the merged TE topology, selected a TE path to be taken by the requested transport service connection as shown on the upper part of Figure 24.

The multi-domain transport service set up coordination includes:

- o Splitting selected for the transport service TE path(s) into segments - one set of segments per each domain involved in the service setup;
- o Issuing a configuration request to each of the involved DCs to set up the transport service across the respective domain. Note that the connection configured paths are required to be expressed in terms of respective abstract TE topologies as exposed to HC by DCs (see lower part of Figure 24).

- o Waiting for the set up complete confirmation from each of the involved DCs. In case one of the DCs reports a failure, HC is responsible to carry out the cleanup/rollback procedures by requesting all involved DCs to tear down the successfully created segments

Figure 24. Transport Service Placement Based on Abstract TE Topology

While processing the received from HC configuration request to set up the transport service, each DC is expected to carry out the transport service mapping procedures (as described in 1.8) resulting in the set up of all the necessary underlay TE tunnels, as well as one or more connections supporting the transport service. As a result, the requested transport service will be provisioned as shown in Figure 25.

- o In the example above the TE tunnel segments that each DC has to set up are the head TE tunnel segment (for domain 1) and the tail TE tunnel segment (for domain 2). For head TE tunnel segment HC can specify in the configuration request only the source TTP (located in node s3 in the example), but not the tunnel's destination TTP, because it is outside of the domain controlled by the DC.

Therefore, the outbound hand-off point (in the form of outbound inter-domain TE link ID/label pair) of each connection segment supporting a TE tunnel non-tail segment (such as head or transit tunnel segment) is expected to be found at the end of the route-object-include-exclude list of the explicit-route-objects configured for that connection segment.

- o Likewise, the inbound hand-off point (in the form of inbound inter-domain TE link ID/label pair) of each connection segment supporting a TE tunnel non-head segment (such as tail or transit tunnel segment) is expected to be found at the beginning of the route-object-include-exclude list of the explicit-route-objects configured for that connection segments.
- o For example, in the figure above the HC can specify the outbound hand-off point of the primary path supporting the head TE tunnel segment. The configuration is to be in the form of the pair of the TE link ID, identifying the inter domain link terminating on node s7, and of the TE label used on that link.
- o In case (not present in this example) we need to setup a Transit Tunnel Segment since the endpoints of the E2E Tunnel are both outside the domain controlled by that DC, the HC would not specify any source or destination TTP (i.e., it would leave the source, destination, src-tp-id and dst-tp-id attributes empty) and it would use the route-object-include-exclude list of the explicit-route-objects to specify the inbound and outbound hand-off points of each connection segment supporting the Transit Tunnel Segment.

The multi-domain transport service tear down coordination entails issuing to each of the involved DCs a configuration request to delete the transport service in the controlled by the DC domain. DCs are expected in this case to release all network resources allocated for the transport service.

The multi-domain transport service modify coordination implies issuing to each of the involved DCs a configuration request to replace the transport service connections according to the newly provided paths and/or modify the connection parameters according to the newly provided configuration.

Figure 25. Multi-domain transport service is provisioned

2. Use Cases

2.1. Use Case 1. Transport service control on a single layer multi-domain transport network

Configuration (Figure 26):

- o Three-domain multi-vendor ODUk/Och transport network;
- o The domains are interconnected via ODUk inter-domain links;

- o Each of the domains is comprised of ODUk/Och network elements (switches) from a separate vendor and is controlled by a single (vendor specific) T-SDN Domain Controller (DC);
- o All DCs expose IETF TE Topology and TE Tunnel model based NBIs;
- o The transport network as a whole is controlled by a single hierarchical T-SDN controller (HC);
- o HC makes use of the NBIs to set up client-provider relationship with each of the DCs and controls via the DCs their respective network domains;
- o Three customer IP/MPLS sites are connected to the transport network via ODUk access links;
- o The customer IP/MPLS routers and the router transport ports connecting the routers to the transport network are managed autonomously and independently from the transport network.

Figure 26 Three-domain ODUk/Och transport network with ODUk access and inter-domain links

Objective: Set up/manipulate/delete a shortest delay unprotected or protected transport service to provide connectivity between customer network elements C-R2 and C-R5

1) TE Topology discovery

All DCs provide to HC respective domain ODUk layer abstract TE topologies. Let's assume that each such topology is a single-node TE topology (as described in 1.3.1, abstract TE topology of this type represents the entire domain as a single asymmetrical/blocking TE node). Let's further assume that the abstract TE nodes representing the domains are attributed with detailed connectivity matrices optimized according to the shortest delay criterion. [Note: single-node abstract TE topologies are assumed for simplicity sake. Alternatively, any DC could have provided an abstract TE topology of any type described in 1.3].

HC merges the provided TE topologies into its own native TE topology (the TE topology merging procedures are discussed in 1.4). The merged TE topology, as well as the TE topologies provided by DCs, are depicted in Figure 27. The merged TE topology homogeneously describes the entire transport network and hence is suitable for path computations across the network. Note that the dotted lines in the Figure connecting the topology access TE links with customer devices illustrate that HC in this use case has neither control nor information on the customer devices/ports and, therefore, can only provide a connectivity between the requested transport service ingress and egress access links (on assumption that the customer transport ports are provisioned independently)

Figure 27. Three-domain single layer transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 27, upper part) HC selects one or more optimal and sufficiently disjoint from each other TE path(s) for the requested transport service connection(s). Resulting TE paths for the requested end-to-end protected transport service, for example, could be as marked on the upper part of Figure 28.

It is important to keep in mind that HC's path computer is capable of performing the necessary path selection only as long as the merged TE topology provides the necessary TE visibility for the path selection, both intra-domain (e.g. by virtue of provided by the abstract TE nodes detailed connectivity matrices) and inter-domain (because of provided inter-domain TE link attributes). In case one or more DCs is/are not capable of or willing to provide the detailed connectivity matrices (that is, DCs expose the respective domains as black boxes - unconstrained TE nodes terminating the inter-domain TE links), HC will not be able to select the end-to-end TE path(s) for the requested transport service on its own. In such a case HC may opt for making use of the Path Computation NBI, exposed by the DCs to explore/evaluate intra-domain TE path availability in real time. IETF TE Tunnel model supports the Path Computation NBI by allowing for the configuration of transport services in COMPUTE_ONLY mode. In this mode the provider is expected to compute TE paths for a requested transport service connections and return the paths in the request's response without triggering the connection provisioning in the network.

Consider, for example, the case when none of the DCs has provided the detailed connectivity matrix attribute for the abstract TE nodes representing the respective domain. In such a case HC may:

1. Request the ingress domain DC (i.e. DC1) to compute intra-domain TE paths connecting the ingress access TE link (i.e. the link facing C-R2) with each of the inter-domain TE links (i.e. links connecting Domain 1 to Domain 2 and Domain 3 respectively);
2. Grow the TE paths returned by DC1 in (1) over the respective outbound inter-domain TE links;
3. Request the neighboring DC(s) (e.g. DC3) to compute all intra-domain TE paths connecting across the domain all inbound into the domain inter-domain TE links reached by the path growing process in (2) with all other (outbound) domain's inter-domain TE links;

4. Augment the TE paths produced in step (2) with the TE paths determined in step (3);
5. Repeat steps (2), (3) and (4) until the resulting TE paths reach the egress domain (i.e. Domain 2);
6. Request the egress domain DC (i.e. DC2) to grow each of the TE paths across the domain to connect them to the egress access TE link (i.e. the link facing C-R5);
7. Select one (or more) most optimal and sufficiently disjoint from each other TE path(s) from the list produced in step (6).

[Note: The transport service path selection method based on Path Computation NBIs exposed by DCs does not scale well and the more domains comprise the network and the more inter-domain links interconnect them, the worse the method works. Realistically, this approach will not work sufficiently well for the networks with more than 3 domains]

Figure 28. TE paths computed for the protected transport service

3) Transport service setup coordination

HC carries out the multi-domain transport service setup coordination as described in 1.9. In particular, HC splits the computed TE path(s) into 3 sets of TE path segments - one set per domain (as shown on the lower part of Figure 28), and issues a TE tunnel configuration request to each of the DCs to set up the requested transport service across the domain under the DC's control. The primary (and secondary) connection explicit path(s) is/are specified in the requests in terms of respective domain abstract TE topologies.

While processing the configuration request, each DC performs the transport service mapping (as described in 1.8). In particular, the DC translates the specified explicit path(s) from abstract into native TE topology terms, sets up supporting underlay TE tunnels

(e.g. OCh TE tunnels), and, then, allocates required ODUk containers on the selected links and provisions the ODUk cross-connects on the switches terminating the links.

If the setup is successfully completed in all three domains, the transport service connection(s) will be provisioned as depicted in Figure 29. If one of the DCs fails to set up its part, all successfully provisioned segments will be asked by HC to be released.

4) Transport service teardown coordination

HC issues to each of DCs a configuration request to release the transport service over the controlled domain, as well as the server layer TE tunnels supporting dynamically created links.

Figure 29. Transport service is provisioned

2.2. Use Case 2. End-to-end TE tunnel control on a single layer multi-domain transport network

Configuration (Figure 26): the same as in use case 1, except that HC in this use case controls customer devices/ports by extracting information from and pushing configuration to the customer site SDN controller(s) managing the customer devices directly.

Objective: Set up//delete an unprotected shortest delay TE tunnel interconnecting end-to-end C-R2 and C-R5

1) TE Topology discovery

As in use case 1 all DCs provide to HC domain ODUk layer abstract TE topologies. Additionally in this use the three customer site controllers expose the TE Topology and Tunnel model based NBIs to HC. Using the TE Topology NBI each customer controller provides to HC the respective customer site domain abstract TE topology. Customer site abstract TE topologies contain abstract TE nodes representing the devices which are directly connected to the transport network. Said abstract TE nodes host TE tunnel termination points, TTPs, representing the ports over which the customer devices are connected to the transport network, and terminate access TE links the TTPs are accessible from (see Figure 30).

Figure 30. Abstract TE topologies provided by all network domains and customer sites

HC merges the provided topologies into its own native TE Topology (the TE topology merging procedures are discussed in 1.4). The merged TE topology is depicted in Figure 31. It homogeneously describes end-to-end not only the entire transport network, but also the customer sites connected to the network and hence is suitable for TE tunnel end to end path computations.

Figure 31. Abstract TE topology describing transport network and connected to it customer sites

2) TE tunnel path computation

Using the merged TE topology (Figure 31) HC selects an optimal TE path for the requested TE tunnel connecting end-to-end the specified TE tunnel termination points, TTPs. The resulting TE path, for example, could be as marked on the upper part of Figure 32.

Figure 32. TE path computed for the TE tunnel

3) TE tunnel setup coordination

HC carries out the multi-domain TE tunnel setup coordination as described for use case 1, except that in this use case HC additionally initiates and controls the setup of the TE tunnel's head and tail segments on the respective customer sites. Note that the customer site controllers behave exactly as transport network domain DCs. In particular, they receive issued by HC configuration requests to set up the TE tunnel's head and tail segments respectively. While processing the requests the customer site controllers perform the necessary provisioning of the TE tunnel's source and destination termination points, as well as of the local sides of the selected

access links. If all segments are successfully provisioned on customer sites and network domains, the TE tunnel connection will be provisioned as marked in Figure 33.

4) TE tunnel teardown coordination

HC issues to each of DCs and customer site controllers a configuration request to release respective segments of the TE tunnel, as well as the server layer TE tunnels supporting dynamically created links.

Figure 33. TE tunnel is provisioned

2.3. Use Case 3. Transport service control on a ODUk/Och multi-domain transport network with Ethernet access links

Configuration (Figure 34): the same as in use case 1, except that all access links in this use case are Ethernet layer links (depicted as

blue lines in the Figure), while all inter-domain links remain to be ODUk layer links.

Figure 34. Three-domain ODUk/Och transport network with Ethernet layer access links

Objective: Set up/delete an unprotected shortest delay transport service supporting connectivity between C-R2 and C-R5

1) TE Topology discovery

In order to make possible for the necessary in this use case multi-layer path computation, each DC exposes to HC two (ODUk layer and Ethernet layer) abstract TE topologies, Additionally, the lower layer (ODUk) TE nodes announce hosted by them TE tunnel termination points, TTPs, capable of adopting the payload carried over the Ethernet layer access links, From the TE Topology model point of view this means that said TTPs are attributed with TE inter-layer locks

matching ones attributed to Ethernet TE links (i.e. TE links provided within Ethernet layer abstract TE topologies).

Ethernet and ODUk layer single node abstract TE topologies catered to HC by each of the DCs are presented in Figure 35.

HC merges the provided TE topologies into its own native TE Topology (the merging procedures are described in 1.4). Importantly in this case HC locks the provided TE topologies not only horizontally, but vertically as well, thus producing a two-layer TE topology homogenously describing both layers of the entire transport network, as well as the client-server layer adaptation relationships between the two layers. This makes the merged TE topology suitable for multi-layer/inter-layer multi-domain transport service path computations. The merged TE topology is presented in Figure 36.

Figure 35. ODUk and Ethernet layer abstract TE topologies exposed by DCs

Figure 36. Two-layer three-domain transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 36) HC selects an optimal TE path for the requested transport service.

Note that if HC's path computer considered only Ethernet layer TE nodes and links, the path computation would fail. This is because the Ethernet layer TE nodes (i.e. D1-e, D2-e and D3-e in the Figure) are disconnected from each other. However, the inter-layer associations (in the form of the TE inter-layer links) make possible for the path computer to select TE path(s) in the lower (ODUk) layer that can be used to set up hierarchy TE tunnel(s) supporting additional dynamic TE link(s) in the upper (Ethernet) layer in order for the requested transport service path computation to succeed.

Let's assume that the resulting TE path is as marked in Figure 37. The red line in the Figure marks the TE path selected for the ODUk layer hierarchy TE tunnel supporting the required Ethernet layer dynamic TE link.

Figure 37. Multi-layer TE path computed for the transport service

3) Transport service setup coordination

HC sets up the requested Ethernet layer transport service in two stages. First, it coordinates the end-to-end setup of the ODUk layer hierarchy TE tunnel between the selected TTPs. If this operation succeeds, a new Ethernet layer dynamic TE link (blue line connecting TE nodes D1-e and D2-e in Figure 38) is automatically added to the merged abstract TE topology. Importantly, as a part of the hierarchy transport service setup both DC1 and DC 2 add a new open-ended Ethernet layer inter-domain dynamic TE link to their respective abstract TE topologies. Second, HC coordinates the setup of the requested (Ethernet layer) transport service. The required TE path for the second stage is marked as fat blue line in the Figure. Note that DC3 controlling domain 3 is only involved in the first stage, but is oblivious to the second stage.

Figure 38. A new Ethernet layer TE link supported by ODUk layer TE tunnel is added to the provided and merged abstract TE topologies

IF all involved DCs confirm successful setup completion, the requested transport service, as well as the supporting server layer hierarchy TE tunnel, will be provisioned as depicted in Figure 39. If one of the DCs fails to set up its segment in either of the layers, all successfully provisioned segments will be requested by HC to be released.

Figure 39. Ethernet transport service and supporting ODUk TE tunnel are provisioned

4) Transport service teardown coordination

First, HC issues to DC1 and DC2 a configuration request to release the Ethernet layer transport service in the respective domains. After that, all three DCs are requested to release the segments of the supporting ODUk layer hierarchy TE tunnel. While processing the request DC1 and DC2 also remove the dynamic Ethernet layer TE links supported by the respective hierarchy TE tunnel's segments, thus the

network's abstract TE topologies are reverted back to the state as shown in Figures 35 and 36.

2.4. Use Case 4. Transport service control on a ODUk/Och multi-domain transport network with multi-function access links

Configuration (Figure 40): the same as in use case 3, except that all access links in this use case are multi-function links (depicted in the Figure as blue compound lines). Let's assume that, depending on configuration, the multi-function access links in this use case can carry either Ethernet or SDH/STM16 layer payload.

Objective: Set up/delete an unprotected shortest delay SDH/STM16 layer transport service interconnecting C-R2 and C-R5

Figure 40. Three-domain ODUk/Och transport network with multi-function access links

1) TE Topology discovery

The TE Topology model considers multi-function links as parallel mutually exclusive TE links each belonging to a separate layer network. For this use case each DC exposes to HC three (ODUk-, Ethernet- and SDH/STM16-layer) abstract TE topologies (generally speaking, one abstract TE topology per each layer network supported by at least one access or inter-domain link). Like in use case 3, the lower layer (ODUk) TE nodes announce hosted by them TE tunnel termination points, TTPs, capable in this case of adopting Ethernet, SDH/STM16 or both layer payloads, The TTPs are attributed with TE inter-layer links matching ones specified for Ethernet and/or SDH/STM16 TE links.

Ethernet, SDH/STM16 and ODUk layer single-node abstract TE topologies catered to HC by each of the DCs are presented in Figure 41.

HC merges the provided topologies into its own native TE Topology (the merging procedures are described in 1.4). As in use case 3 HC locks the provided TE topologies not only horizontally (i.e. between domains), but vertically (between layers) as well, thus producing a three-layer TE topology homogenously describing the three layers of the entire transport network, as well as the client-server layer adaptation relationships between the layers. This makes the merged TE topology suitable for multi-layer/inter-layer multi-domain transport service path computations. The merged TE topology is presented in Figure 42.

Figure 41. ODUk, Ethernet and SDH/STM16 layer abstract TE topologies exposed by DCs

Figure 42. Three-layer three-domain transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 42) HC's path computer selects a TE path for the requested transport service. For example, for the SDH/STM16 layer unprotected transport service the resulting TE path could be determined as marked in Figure 43.

Figure 43. Multi-layer TE path computed for SDH/STM16 layer transport service

3) Transport service setup coordination

Same as in use case 3.

4) Transport service teardown coordination

Same as in use case 3.

2.5. Use Case 5. Real time updates of IP/MPLS layer TE link attributes that depend on supporting transport connectivity (e.g. transport SRLGs, propagation delay, etc.)

Configuration (Figure 26): the same as in use case 1,

Objective: A transport service interconnecting transport ports of two IP routers across a transport network is likely to serve a link in IP/MPLS layer network, which is usually controlled by a client of the

transport network, such as IP/MPLS Controller. Performance of TE applications (e.g. path computer) running on the IP/MPLS Controller depends on the accuracy of IP/MPLS layer TE link attributes. Some of these attributes can change over time and are known real-time only to a transport network controller, such as HC. Examples of said attributes are transport SRLGs, propagation delay metric, protection capacities and status, etc. The objective of this use case is to ensure up-to-date state of said attributes in the IP/MPLS Controller's internal TED via necessary updates provided in a timely manner by the controller (e.g. HC) managing transport connectivity supporting IP/MPLS layer links.

Realization:

- o HC exposes and supports IETF TE Topology and TE Tunnel model based NBIs (the same NBIs that are exposed by DCs serving HC);
- o IP/MPLS Controller makes use of the exposed NBIs to set up the respective client-provider relationships with HC;
- o IP/MPLS Controller uses the TE Tunnel NBI to configure with HC a transport service interconnecting transport ports of a pair of IP routers desired to be adjacent in the IP/MPLS layer network. The TE Tunnel model allows for specifying in the transport service configuration request the TE topology and link IDs of the IP/MPLS TE link the requested transport service will be serving;
- o IP/MPLS Controller uses the TE Topology NBI to subscribe with HC on the IP/MPLS TE link notifications with respect to changes in the TE link's attributes, such as SRLGs, propagation delay, protection capabilities/status, etc.;
- o HC uses the TE Topology NBI to convey the requested notifications when HC learns the attributes IP/MPLS has expressed interest in or detects any changes since previous notifications (for example, due to network failure restoration/reversion procedures happened to the transport connectivity that supports the failure affected IP/MPLS links)

2.6. Use Case 6. Virtual Network Service

Configuration (Figure 26): the same as in use case 1,

Objective: Set up two Virtual Networks for the client, with Virtual Network 1 interconnecting customer IP routers C-R1, C-R7 and C-R4 over a single-node abstract TE topology, and Virtual Network 2

interconnecting customer IP routers C-R2, C-R3, C-R8, C-R5 and C-R6 over a full mesh link abstract TE topology as depicted in Figure 44.

[Note: A client of a transport network may want to limit the transport network connectivity of a particular type and quality within distinct subsets of its network elements interconnected across the transport network. Furthermore, a given transport network may serve more than one client. In this case some or all clients may want to ensure the availability of transport network resources in case dynamic (re-)connecting of their network elements across the transport network is envisioned. In all such cases a client may want to set up one or more Virtual Networks over provided transport network]

1) Virtual Network setup

From the client's point of view a Virtual Network setup includes the following procedures:

- o Identifying the Virtual Network membership - a subset of the client's network elements/ports to be interconnected over the abstract TE topology configured for the Virtual Network. Note that from the transport network provider's point of view this effectively determines the list of abstract TE topology's open-ended access TE links;
- o Deciding on the Virtual Network's abstract TE topology type (e.g. single-node vs. link mesh), optimization criterion (e.g. shortest delay vs. smallest cost), bandwidth, link disjointedness, adaptation capabilities and other requirements/constraints, as well as, whether the TE tunnels supporting the abstract TE topology need to be pre-established or established on demand (i.e. when respective abstract TE topology elements are selected for a client transport service);
- o Using the IETF TE Topology model based NBI exposed by the transport network controller (i.e. HC), configure the Virtual Network's abstract TE topology. Let's assume that in this use case the abstract TE topology for Virtual Network 1 is configured as a single-node abstract TE topology (see section 1.3.1) with the abstract TE node's detailed connectivity matrix optimized according to the shortest delay criteria. Likewise, the abstract TE topology for Virtual Network 2 is configured as a full-mesh link abstract TE topology (see section 1.3.2) optimized according to the smallest cost criteria with each of the abstract TE links to be supported by pre-established end-to-end protected TE tunnels.

[Note: Virtual Network's abstract TE topology (re-)configuration/negotiation process is no different from one that happens, for example, between HC and its providers, DCs, and is described in section 1.5]

Figure 44. Virtual Networks provided for a transport network client

2) Using Virtual Network

Recall that use case 1 was about setting up a transport service interconnecting customer network elements C-R2 and C-R5 across the transport network. With the Virtual Network 2 in place, the client could have used the Virtual Network's TE topology to select a TE path for the service. The TE Tunnel model based NBI allows for the client to specify the Virtual Network's TE topology ID, as well, as the selected TE path (for example, as marked in Figure 45) as a configured path attribute in the transport service configuration request to ensure that the intended transport network resources are used for the service.

Figure 45. Transport service TE path is selected on Virtual Network's TE topology

3. Security Considerations

This document does not define networking protocols and data, hence are not directly responsible for security risks.

4. IANA Considerations

This document has no actions for IANA.

5. References

5.1. Normative References

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

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo-15 (work in progress), February 2018.

[I-D.ietf-teas-yang-te]

Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-13 (work in progress), March 2018.

5.2. Informative References

[RFC2702] Awduche, D., "Requirements for Traffic Engineering Over MPLS", RFC 2702, September 1999.

6. Acknowledgments

TBD.

Appendix A.

Data Examples

This section contains examples of an instance data in the JSON encoding [RFC7951].

A.1. Use Case 1

In the use case described in Section 2.1. , there are three provider network domains, each of them is represented as an abstract TE topology. The JSON encoded example data configurations for the three domains are:

A.1.1. Domain 1

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {}
        },
        "network-id": "otn-domain1-abs",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:otn-domain1-abs",
        "node": [
          {
            "node-id": "D1",
            "te-node-id": "2.0.1.1",
            "te": {
              "te-node-attributes": {
                "domain-id" : 1,
                "is-abstract": [null],
                "underlay-topology": "domain1-och",
                "connectivity-matrices": {
                  "is-allowed": true,
                  "path-constraints": {
                    "bandwidth-generic": {
                      "te-bandwidth": {
                        "otn": [
                          {
                            "rate-type": "odul",
                            "counter": 2
                          }
                        ]
                      }
                    }
                  }
                }
              }
            }
          ]
        }
      ]
    }
  }
}
```



```
    }
  ]
}
}
}
"connectivity-matrix": [
{
  "id": 10302,
  "from": "1-0-3",
  "to": "1-2-0"
},
{
  "id": 10203,
  "from": "1-0-2",
  "to": "1-3-0"
},
{
  "id": 10311,
  "from": "1-0-3",
  "to": "1-11-0"
},
{
  "id": 11103,
  "from": "1-0-11",
  "to": "1-3-0"
},
{
  "id": 10903,
  "from": "1-0-9",
  "to": "1-3-0"
},
{
  "id": 10309,
  "from": "1-0-3",
  "to": "1-9-0"
},
{
  "id": 10910,
  "from": "1-0-9",
  "to": "1-10-0"
},
},
},
```

```

        {
            "id": 11009,
            "from": "1-0-10",
            "to": "1-9-0"
        },
        {
            "id": 20910,
            "from": "1-1-9",
            "to": "1-10-0"
        },
        {
            "id": 21009,
            "from": "1-0-10",
            "to": "1-9-1"
        },
        {
            "id": 20911,
            "from": "1-1-9",
            "to": "1-11-0"
        },
        {
            "id": 21109,
            "from": "1-0-11",
            "to": "1-9-1"
        }
    ]
}
},
"termination-point": [
    {
        "tp-id": "1-0-3",
        "te-tp-id": 10003
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk"
                }
            ]
        }
    }
]
}

```

```
    },
    {
      "tp-id": "1-3-0",
      "te-tp-id": 10300
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    }
  ],
  {
    "tp-id": "1-0-9",
    "te-tp-id": 10009
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
],
{
  "tp-id": "1-9-0",
  "te-tp-id": 10900
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
],
{
  "tp-id": "1-1-9",
  "te-tp-id": 10109
  "te": {
```

```
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      },
    },
    {
      "tp-id": "1-9-1",
      "te-tp-id": 10901
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    },
    {
      "tp-id": "1-0-2",
      "te-tp-id": 10002
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    },
    {
      "tp-id": "1-2-0",
      "te-tp-id": 10200
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    }
  ]
}
```

```
    ]
  },
  {
    "tp-id": "1-0-10",
    "te-tp-id": 10010
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-10-0",
    "te-tp-id": 11000
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-0-11",
    "te-tp-id": 10011
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-11-0",
```

```

        "te-tp-id": 11100
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk"
                }
            ]
        }
    },
    {
        "tp-id": "1-1-11",
        "te-tp-id": 10111
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk"
                }
            ]
        }
    },
    {
        "tp-id": "1-11-1",
        "te-tp-id": 11101
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk"
                }
            ]
        }
    }
]
}

```

A.1.2. Domain 2

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {}
        },
        "network-id": "otn-domain2-abs",
        "provider-id": 202,
        "client-id": 300,
        "te-topology-id": "te-topology:otn-domain2-abs",
        "node": [
          {
            "node-id": "D2",
            "te-node-id": "2.0.2.2",
            "te": {
              "te-node-attributes": {
                "is-abstract": [null],
                "underlay-topology": "domain2-och",
                "connectivity-matrices": {
                  "is-allowed": true,
                  "path-constraints": {
                    "bandwidth-generic": {
                      "te-bandwidth": {
                        "otn": [
                          {
                            "rate-type": "odul",
                            "counter": 2
                          }
                        ]
                      }
                    }
                  }
                }
              }
            }
          ]
        },
        "connectivity-matrix": [
          {
            "id": 12125,
            "from": "1-0-21",
            "to": "1-25-0"
          }
        ],
      }
    ]
  }
}
```

```
{
  "id": 12521,
  "from": "1-0-25",
  "to": "1-21-0"
},
{
  "id": 12128,
  "from": "1-0-21",
  "to": "1-28-0"
},
{
  "id": 12821,
  "from": "1-0-28",
  "to": "1-21-0"
},
{
  "id": 12231,
  "from": "1-0-22",
  "to": "1-31-0"
},
{
  "id": 13122,
  "from": "1-0-31",
  "to": "1-22-0"
},
{
  "id": 22228,
  "from": "1-1-22",
  "to": "1-28-0"
},
{
  "id": 22822,
  "from": "1-0-28",
  "to": "1-22-1"
},
{
  "id": 12528,
  "from": "1-0-25",
  "to": "1-28-0"
},
{
```



```
        "id": 12825,  
        "from": "1-0-28",  
        "to": "1-25-0"  
      }  
    ]  
  }  
},  
"termination-point": [  
  {  
    "tp-id": "1-0-21",  
    "te-tp-id": 10021  
    "te": {  
      "interface-switching-capability": [  
        {  
          "switching-capability": "switching-otn",  
          "encoding": "lsp-encoding-oduk"  
        }  
      ]  
    }  
  },  
  {  
    "tp-id": "1-21-0",  
    "te-tp-id": 12100  
    "te": {  
      "interface-switching-capability": [  
        {  
          "switching-capability": "switching-otn",  
          "encoding": "lsp-encoding-oduk"  
        }  
      ]  
    }  
  },  
  {  
    "tp-id": "1-0-22",  
    "te-tp-id": 10022  
    "te": {  
      "interface-switching-capability": [  
        {  
          "switching-capability": "switching-otn",  
          "encoding": "lsp-encoding-oduk"
```

```

    }
  ]
}
},
{
  "tp-id": "1-22-0",
  "te-tp-id": 12200
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "1-1-22",
  "te-tp-id": 10122
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "1-22-1",
  "te-tp-id": 12201
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{

```

```
"tp-id": "1-0-25",
"te-tp-id": 10025
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
}
},
{
  "tp-id": "1-25-0",
  "te-tp-id": 12500
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "1-1-25",
  "te-tp-id": 10125
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
},
{
  "tp-id": "1-25-1",
  "te-tp-id": 12501
  "te": {
    "interface-switching-capability": [
      {
```

```
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  },
  {
    "tp-id": "1-0-28",
    "te-tp-id": 10028
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-28-0",
    "te-tp-id": 12800
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-0-31",
    "te-tp-id": 10031
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
}
```



```
"underlay-topology": "domain3-och",
"connectivity-matrices": {
  "is-allowed": true,
  "path-constraints": {
    "bandwidth-generic": {
      "te-bandwidth": {
        "otn": [
          {
            "rate-type": "odul",
            "counter": 2
          }
        ]
      }
    }
  }
}
"connectivity-matrix": [
  {
    "id": 13638,
    "from": "1-0-38",
    "to": "1-38-0"
  },
  {
    "id": 13836,
    "from": "1-0-38",
    "to": "1-36-0"
  },
  {
    "id": 13639,
    "from": "1-0-36",
    "to": "1-39-0"
  },
  {
    "id": 13936,
    "from": "1-0-39",
    "to": "1-36-0"
  },
  {
    "id": 23636,
    "from": "1-0-36",
    "to": "1-36-1"
  },
]
```

```

        {
            "id": 33636,
            "from": "1-1-36",
            "to": "1-36-0"
        },
        {
            "id": 13739,
            "from": "1-0-37",
            "to": "1-39-0"
        },
        {
            "id": 13937,
            "from": "1-0-39",
            "to": "1-37-0"
        },
        {
            "id": 23737,
            "from": "1-0-37",
            "to": "1-37-1"
        },
        {
            "id": 33737,
            "from": "1-1-37",
            "to": "1-37-0"
        }
    ]
}
},
"termination-point": [
    {
        "tp-id": "1-0-36",
        "te-tp-id": 10036
        "te": {
            "interface-switching-capability": [
                {
                    "switching-capability": "switching-otn",
                    "encoding": "lsp-encoding-oduk"
                }
            ]
        }
    }
]
}

```

```
    },
    {
      "tp-id": "1-36-0",
      "te-tp-id": 13600
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    }
  ],
  {
    "tp-id": "1-0-37",
    "te-tp-id": 10037
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  }
],
{
  "tp-id": "1-37-0",
  "te-tp-id": 13700
  "te": {
    "interface-switching-capability": [
      {
        "switching-capability": "switching-otn",
        "encoding": "lsp-encoding-oduk"
      }
    ]
  }
}
],
{
  "tp-id": "1-1-37",
  "te-tp-id": 10137
  "te": {
```



```
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      },
    },
    {
      "tp-id": "1-37-1",
      "te-tp-id": 13701
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    },
    {
      "tp-id": "1-0-39",
      "te-tp-id": 10039
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    },
    {
      "tp-id": "1-39-0",
      "te-tp-id": 13900
      "te": {
        "interface-switching-capability": [
          {
            "switching-capability": "switching-otn",
            "encoding": "lsp-encoding-oduk"
          }
        ]
      }
    }
  ]
}
```

```
    ]
  },
  {
    "tp-id": "1-0-36",
    "te-tp-id": 10036
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-36-0",
    "te-tp-id": 13600
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-0-38",
    "te-tp-id": 10038
    "te": {
      "interface-switching-capability": [
        {
          "switching-capability": "switching-otn",
          "encoding": "lsp-encoding-oduk"
        }
      ]
    }
  },
  {
    "tp-id": "1-38-0",
```

```
"te-tp-id": 13800
"te": {
  "interface-switching-capability": [
    {
      "switching-capability": "switching-otn",
      "encoding": "lsp-encoding-oduk"
    }
  ]
}
}
```

Authors' Addresses

Igor Bryskin
Huawei Technologies
Email: Igor.Bryskin@huawei.com

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

Tarek Saad
Cisco Systems Inc
Email: tsaad@cisco.com

Xufeng Liu
Volta Networks
Email: xufeng.liu.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 6, 2018

I. Bryskin
Huawei Technologies
X. Liu
Jabil
J. Guichard
Y. Lee
Huawei Technologies
L. Contreras
Telefonica
D. Ceccarelli
Ericsson
J. Tantsura
Nuage Networks
June 4, 2018

Use Cases for SF Aware Topology Models
draft-ietf-teas-use-cases-sf-aware-topo-model-00

Abstract

This document describes some use cases that benefit from the network topology models that are service and network function aware.

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 December 6, 2018.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
2. Terminology	3
3. Exporting SF/NF Information to Network Clients and Other Network SDN Controllers	4
4. Flat End-to-end SFCs Managed on Multi-domain Networks	5
5. Managing SFCs with TE Constraints	6
6. SFC Protection and Load Balancing	7
7. Network Clock Synchronization	10
8. Client - Provider Network Slicing Interface	11
9. Dynamic Assignment of Regenerators for L0 Services	11
10. Dynamic Assignment of OAM Functions for L1 Services	12
11. SFC Abstraction and Scaling	13
12. Dynamic Compute/VM/Storage Resource Assignment	13
13. Application-aware Resource Operations and Management	14
14. IANA Considerations	15
15. Security Considerations	15
16. Acknowledgements	15
17. References	15
17.1. Normative References	16
17.2. Informative References	16
Authors' Addresses	17

1. Introduction

Normally network connectivity services are discussed as a means to inter-connect various abstract or physical network topological elements, such as ports, link termination points and nodes [I-D.ietf-teas-yang-te-topo] [I-D.ietf-teas-yang-te]. However, the connectivity services, strictly speaking, interconnect not the network topology elements per-se, rather, located on/associated with the various network and service functions [RFC7498] [RFC7665]. In many scenarios it is beneficial to decouple the service/network functions from the network topology elements hosting them, describe them in some unambiguous and identifiable way (so that it would be possible, for example, to auto-discover on the network topology service/network functions with identical or similar functionality and characteristics) and engineer the connectivity between the service/network functions, rather than between their current topological

locations. The purpose of this document is to describe some use cases that could benefit from such an approach.

2. Terminology

- o Network Function (NF): A functional block within a network infrastructure that has well-defined external interfaces and well-defined functional behaviour [ETSI-NFV-TERM]. Such functions include message router, CDN, session border controller, WAN acceleration, DPI, firewall, NAT, QoE monitor, PE router, BRAS, and radio/fixed access network nodes.
- o Network Service: Composition of Network Functions and defined by its functional and behavioural specification. The Network Service contributes to the behaviour of the higher layer service, which is characterized by at least performance, dependability, and security specifications. The end-to-end network service behaviour is the result of the combination of the individual network function behaviours as well as the behaviours of the network infrastructure composition mechanism [ETSI-NFV-TERM].
- o Service Function (SF): A function that is responsible for specific treatment of received packets. A service function can act at various layers of a protocol stack (e.g., at the network layer or other OSI layers). As a logical component, a service function can be realized as a virtual element or be embedded in a physical network element. One or more service functions can be embedded in the same network element. Multiple occurrences of the service function can exist in the same administrative domain. A non-exhaustive list of service functions includes: firewalls, WAN and application acceleration, Deep Packet Inspection (DPI), server load balancers, NAT44 [RFC3022], NAT64 [RFC6146], HTTP header enrichment functions, and TCP optimizers. The generic term "L4-L7 services" is often used to describe many service functions [RFC7498].
- o Service Function Chain (SFC): A service function chain defines an ordered or partially ordered set of abstract service functions and ordering constraints that must be applied to packets, frames, and/or flows selected as a result of classification. An example of an abstract service function is a firewall. The implied order may not be a linear progression as the architecture allows for SFCs that copy to more than one branch, and also allows for cases where there is flexibility in the order in which service functions need to be applied. The term "service chain" is often used as shorthand for "service function chain" [RFC7498].

- o Connectivity Service: Any service between layer 0 and layer 3 aiming at delivering traffic among two or more end customer edge nodes connected to provider edge nodes. Examples include L3VPN, L2VPN etc.
 - o Link Termination Point (LTP): 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 [I-D.ietf-teas-yang-te-topo].
 - o Tunnel Termination Point (TTP): 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 with the TE node scope unique ID. 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 [I-D.ietf-teas-yang-te-topo].
3. Exporting SF/NF Information to Network Clients and Other Network SDN Controllers

In the context of Service Function Chain (SFC) orchestration one existing problem is that there is no way to formally describe a Service or Network Function in a standard way (recognizable/understood by a third party) as a resource of a network topology node.

One implication of this is that there is no way for the orchestrator to give a network client even a ball-park idea as to which network's SFs/NFs are available for the client's use/control and where they are located in the network even in terms of abstract topologies/virtual networks configured and managed specifically for the client. Consequently, the client has no say on how the SFCs provided for the client by the network should be set up and managed (which SFs are to be used and how they should be chained together, optimized, manipulated, protected, etc.).

Likewise, there is no way for the orchestrator to export SF/NF information to other network controllers. The SFC orchestrator may serve, for example, a higher level controller (such as Network Slicing Orchestrator), with the latter wanting at least some level of control as to which SFs/NFs it wants on its SFCs and how the Service Function Paths (SFPs) are to be routed and provisioned, especially, if it uses services of more than one SFC orchestrator.

The issue of exporting of SF/NF information could be addressed by defining a model, in which formally described/recognizable SF/NF

instances are presented as topological elements, for example, hosted by TE, L3 or L2 topology nodes (see Figure 1). The model could describe whether, how and at what costs the SFs/NFs hosted by a given node could be chained together, how these intra-node SFCs could be connected to the node's Service Function Forwarders (SFFs, entities dealing with SFC NSHs and metadata), and how the SFFs could be connected to the node's Tunnel and Link Termination Points (TTPs and LTPs) to chain the intra-node SFCs across the network topology.

The figure is available in the PDF format.

Figure 1: SF/NF aware TE topology

4. Flat End-to-end SFCs Managed on Multi-domain Networks

SFCs may span multiple administrative domains, each of which controlled by a separate SFC controller. The usual solution for such a scenario is the Hierarchical SFCs (H-SFCs), in which the higher level orchestrator controls only SFs located on domain border nodes. Said higher level SFs are chained together into higher level SFCs via lower level (intra-domain) SFCs provisioned and controlled independently by respective domain controllers. The decision as to which higher level SFCs are connected to which lower level SFCs is driven by packet re-classification every time the packet enters a given domain. Said packet re-classification is a very time-consuming operation. Furthermore, the independent nature of higher and lower level SFC control is prone to configuration errors, which may lead to long lasting loops and congestions. It is highly desirable to be

able to set up and manage SFCs spanning multiple domains in a flat way as far as the data plane is concerned (i.e. with a single packet classification at the ingress into the multi-domain network but without re-classifications on domain ingress nodes).

One way to achieve this is to have the domain controllers expose SF/NF-aware topologies, and have the higher level orchestrator operate on the network-wide topology, the product of merging of the topologies catered by the domain controllers. This is similar in spirit to setting up, coordinating and managing the transport connectivity (TE tunnels) on a multi-domain multi-vendor transport network.

5. Managing SFCs with TE Constraints

Some SFCs require per SFC link/element and end-to-end TE constraints (bandwidth, delay/jitter, fate sharing/diversity. etc.). Said constraints could be ensured via carrying SFPs inside overlays that are traffic engineered with the constraints in mind. A good analogy would be orchestrating delay constrained L3 VPNs. One way to support such L3 VPNs is to carry MPLS LSPs interconnecting per-VPN VRFs inside delay constrained TE tunnels interconnecting the PEs hosting the VRFs.

—

Figure 2: L3 VPN with delay constraints

Planning, computing and provisioning of TE overlays to constrain arbitrary SFCs, especially those that span multiple administrative domains with each domain controlled by a separate controller, is a very difficult challenge. Currently it is addressed by pre-provisioning on the network of multiple TE tunnels with various TE characteristics, and "nailing down" SFs/NFs to "strategic" locations (e.g. nodes terminating many of such tunnels) in a hope that an

adequate set of tunnels could be found to carry the SFP of a given TE-constrained SFC. Such an approach is especially awkward in the case when some or all of the SFs/NFs are VNFs (i.e. could be instantiated at multiple network locations).

SF/NF-aware TE topology model in combination with TE tunnel model will allow for the network orchestrator (or a client controller) to compute, set up and manipulate the TE overlays in the form of TE tunnel chains (see Figure 3).

Said chains could be dual-optimized compromising on optimal SF/NF locations with optimal TE tunnels interconnecting them. The TE tunnel chains (carrying multiple similarly constrained SFPs) could be adequately constrained both at individual TE tunnel level and at the chain end-to-end level.

—

Figure 3: SFC with TE constraints

6. SFC Protection and Load Balancing

Currently the combination of TE topology & tunnel models offers to a network controller various capabilities to recover an individual TE tunnel from network failures occurred on one or more network links or transit nodes on the TE paths taken by the TE tunnel's connection(s). However, there is no simple way to recover a TE tunnel from a failure affecting its source or destination node. SF/NF-aware TE topology

model can decouple the association of a given SF/NF with its location on the network topology by presenting multiple, identifiable as mutually substitutable SFs/NFs hosted by different TE topology nodes. So, for example, if it is detected that a given TE tunnel destination node is malfunctioning or has gone out of service, the TE tunnel could be re-routed to terminate on a different node hosting functionally the same SFs/NFs as ones hosted by the failed node (see Figures 6).

This is in line with the ACTN edge migration and function mobility concepts [I-D.ietf-teas-actn-framework]. It is important to note that the described strategy works much better for the stateless SFs/NFs. This is because getting the alternative stateful SFs/NFs into the same respective states as the current (i.e. active, affected by failure) are is a very difficult challenge.

—

Figure 4: SFC recovery: SF2 on node NE1 fails

At the SFC level the SF/NF-aware TE topology model can offer SFC dynamic restoration capabilities against failed/malfunctioning SFs/NFs by identifying and provisioning detours to a TE tunnel chain, so that it starts carrying the SFC's SFPs towards healthy SFs/NFs that are functionally the same as the failed ones. Furthermore, multiple parallel TE tunnel chains could be pre-provisioned for the purpose of SFC load balancing and end-to-end protection. In the latter case

said parallel TE tunnel chains could be placed to be sufficiently disjoint from each other.

—

Figure 5: SFC recovery: SFC SF1-SF2-SF6 is recovered after SF2 on node N1 has failed

—

Figure 6: SFC recovery: SFC SF1-SF2-SF6 is recovered after node N1 has failed

7. Network Clock Synchronization

Many current and future network applications (including 5g and IoT applications) require very accurate time services (PTP level, ns resolution). One way to implement the adequate network clock synchronization for such services is via describing network clocks as NFs on an NF-aware TE topology optimized to have best possible delay variation characteristics. Because such a topology will contain delay/delay variation metrics of topology links and node cross-connects, as well as costs in terms of delay/delay variation of connecting clocks to hosting them node link and tunnel termination points, it will be possible to dynamically select and provision bi-directional time-constrained deterministic paths or trees connecting clocks (e.g. grand master and boundary clocks) for the purpose of exchange of clock synchronization information. Note that network clock aware TE topologies separately provided by domain controllers will enable multi-domain network orchestrator to set up and manipulate the clock synchronization paths/trees spanning multiple network domains.

8. Client - Provider Network Slicing Interface

3GPP defines network slice as "a set of network functions and the resources for these network functions which are arranged and configured, forming a complete logical network to meet certain network characteristics" [I-D.defoy-netslices-3gpp-network-slicing] [_3GPP.28.801]. Network slice could be also defined as a logical partition of a provider's network that is owned and managed by a tenant. SF/NF-aware TE topology model has a potential to support a very important interface between network slicing clients and providers because, on the one hand, the model can describe holistically and hierarchically the client's requirements and preferences with respect to a network slice functional, topological and traffic engineering aspects, as well as of the degree of resource separation/ sharing between the slices, thus allowing for the client (up to agreed upon extent) to dynamically (re-)configure the slice or (re-)schedule said (re-)configurations in time, while, on the other hand, allowing for the provider to convey to the client the slice's operational state information and telemetry the client has expressed interest in.

9. Dynamic Assignment of Regenerators for L0 Services

On large optical networks, some of provided to their clients L0 services could not be provisioned as single OCh trails, rather, as chains of such trails interconnected via regenerators, such as 3R regenerators. Current practice of the provisioning of such services requires configuration of explicit paths (EROs) describing identity and location of regenerators to be used. A solution is highly desirable that could:

- o Identify such services based, for example, on optical impairment computations;
- o Assign adequate for the services regenerators dynamically out of the regenerators that are grouped together in pools and strategically scattered over the network topology nodes;
- o Compute and provision supporting the services chains of optical trails interconnected via so selected regenerators, optimizing the chains to use minimal number of regenerators, their optimal locations, as well as optimality of optical paths interconnecting them;
- o Ensure recovery of such chains from any failures that could happen on links, nodes or regenerators along the chain path.

NF-aware TE topology model (in this case L1 NF-aware L0 topology model) is just the model that could provide a network controller (or even a client controller operating on abstract NF-aware topologies provided by the network) to realize described above computations and orchestrate the service provisioning and network failure recovery operations (see Figure 7).

—

Figure 7: Optical tunnel as TE-constrained SFC of 3R regenerators. Red trail (not regenerated) is not optically reachable, but blue trail (twice regenerated) is

10. Dynamic Assignment of OAM Functions for L1 Services

OAM functionality is normally managed by configuring and manipulating TCM/MEP functions on network ports terminating connections or their segments over which OAM operations, such as performance monitoring, are required to be performed. In some layer networks (e.g. Ethernet) said TCMS/MEPs could be configured on any network ports. In others (e.g. OTN/ODUk) the TCMS/MEPs could be configured on some (but not all network ports) due to the fact that the OAM functionality (i.e. recognizing and processing of OAM messages, supporting OAM protocols and FSMs) requires in these layer networks certain support in the data plane, which is not available on all network nodes. This makes TCMS/MEPs good candidates to be modeled as NFs. This also makes TCM/MEP aware topology model a good basis for placing dynamically an ODUk connection to pass through optimal OAM

locations without mandating the client to specify said locations explicitly.

—

Figure 8: Compute/storage resource aware topology

11. SFC Abstraction and Scaling

SF/NF-aware topology may contain information on native SFs/NFs (i.e. SFs/NFs as known to the provider itself) and/or abstract SFs/NFs (i.e. logical/macro SFs/NFs representing one or more SFCs each made of native and/or lower level abstract SFs/NFs). As in the case of abstracting topology nodes, abstracting SFs/NFs is hierarchical in nature - the higher level of SF/NF-aware topology, the "larger" abstract SFs/NFs are, i.e. the larger data plane SFCs they represent. This allows for managing large scale networks with great number of SFs/NFs (such as Data Center interconnects) in a hierarchical, highly scalable manner resulting in control of very large number of flat in the data plane SFCs that span multiple domains.

12. Dynamic Compute/VM/Storage Resource Assignment

In a distributed data center network, virtual machines for compute resources may need to be dynamically re-allocated due to various reasons such as DCI network failure, compute resource load balancing, etc. In many cases, the DCI connectivity for the source and the destination is not predetermined. There may be a pool of sources and a pool of destination data centers associated with re-allocation of compute/VM/storage resources. There is no good mechanism to date to capture this dynamicity nature of compute/VM/storage resource reallocation. Generic Compute/VM/Storage resources can be described and announced as a SF, where a DC hosting these resources can be modeled as an abstract node. Topology interconnecting these abstract nodes (DCs) in general is of multi-domain nature. Thus, SF-aware

topology model can facilitate a joint optimization of TE network resources and Compute/VM/Storage resources and solve Compute/VM/Storage mobility problem within and between DCs (see Figure 8).

13. Application-aware Resource Operations and Management

Application stratum is the functional grouping which encompasses application resources and the control and management of these resources. These application resources are used along with network services to provide an application service to clients/end-users. Application resources are non-network resources critical to achieving the application service functionality. Examples of application resources include: caches, mirrors, application specific servers, content, large data sets, and computing power. Application service is a networked application offered to a variety of clients (e.g., server backup, VM migration, video cache, virtual network on-demand, 5G network slicing, etc.). The application servers that host these application resources can be modeled as an abstract node. There may be a variety of server types depending on the resources they host. Figure 9 shows one example application aware topology for video cache server distribution.

—

Figure 9: Application aware topology

14. IANA Considerations

This document has no actions for IANA.

15. Security Considerations

This document does not define networking protocols and data, hence is not directly responsible for security risks.

16. Acknowledgements

The authors would like to thank Maarten Vissers, Joel Halpern, and Greg Mirsky for their helpful comments and valuable contributions.

17. References

17.1. Normative References

- [RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <<https://www.rfc-editor.org/info/rfc7498>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [ETSI-NFV-TERM]
ETSI, "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV", ETSI GS NFV 003 V1.2.1, December 2014.
- [I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-15 (work in progress), February 2018.
- [I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-14 (work in progress), March 2018.

17.2. Informative References

- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <<https://www.rfc-editor.org/info/rfc3022>>.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <<https://www.rfc-editor.org/info/rfc6146>>.
- [I-D.ietf-sfc-hierarchical]
Dolson, D., Homma, S., Lopez, D., and M. Boucadair, "Hierarchical Service Function Chaining (hSFC)", draft-ietf-sfc-hierarchical-08 (work in progress), April 2018.

[I-D.defoy-netslices-3gpp-network-slicing]

Foy, X. and A. Rahman, "Network Slicing - 3GPP Use Case", draft-defoy-netslices-3gpp-network-slicing-02 (work in progress), October 2017.

[_3GPP.28.801]

3GPP, "Study on management and orchestration of network slicing for next generation network", 3GPP TR 28.801 V2.0.0, September 2017, <<http://www.3gpp.org/ftp/Specs/html-info/28801.htm>>.

[I-D.ietf-teas-actn-framework]

Ceccarelli, D. and Y. Lee, "Framework for Abstraction and Control of Traffic Engineered Networks", draft-ietf-teas-actn-framework-15 (work in progress), May 2018.

Authors' Addresses

Igor Bryskin
Huawei Technologies

EMail: Igor.Bryskin@huawei.com

Xufeng Liu
Jabil

EMail: xufeng.liu.ietf@gmail.com

Jim Guichard
Huawei Technologies

EMail: james.n.guichard@huawei.com

Young Lee
Huawei Technologies

EMail: leeyoung@huawei.com

Luis Miguel Contreras Murillo
Telefonica

EMail: luismiguel.contrerasmurillo@telefonica.com

Daniele Ceccarelli
Ericsson

EMail: daniele.ceccarelli@ericsson.com

Jeff Tantsura
Nuage Networks

EMail: jefftant.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2019

X. Liu
Volta Networks
I. Bryskin
Huawei Technologies
V. Beeram
Juniper Networks
T. Saad
Cisco Systems Inc
H. Shah
Ciena
O. Gonzalez de Dios
Telefonica
July 1, 2018

YANG Data Model for Layer 3 TE Topologies
draft-ietf-teas-yang-l3-te-topo-02

Abstract

This document defines a YANG data model for layer 3 traffic engineering topologies.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 2, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Terminology	3
1.2. Tree Diagrams	3
2. Modeling Considerations for L3 TE Topologies	3
2.1. Relationship Between Layer 3 Topology and TE topology . .	3
2.2. Relationship Modeling	4
2.2.1. Topology Referencing	4
2.2.2. Node Referencing	4
2.2.3. Link Termination Point Referencing	4
2.2.4. Link Referencing	5
2.3. Topology Type Modeling	5
3. Packet Switching Technology Extensions	5
3.1. Technology Specific Link Attributes	5
3.2. Performance Metric	6
4. Model Structure	6
4.1. Layer 3 TE Topology Module	6
4.2. Packet Switching TE Topology Module	7
5. YANG Modules	21
5.1. Layer 3 TE Topology Module	21
5.2. Packet Switching TE Topology Module	26
6. IANA Considerations	31
7. Security Considerations	33
8. References	35
8.1. Normative References	35
8.2. Informative References	37
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations	38
A.1. Layer 3 TE Topology State Module	38
A.2. Packet Switching TE Topology State Module	41
Authors' Addresses	46

1. Introduction

This document defines a YANG [RFC7950] data model for describing the relationship between a layer 3 network topology [RFC8346] and a TE topology [I-D.ietf-teas-yang-te-topo].

When traffic engineering is enabled on a layer 3 network topology, there will be a corresponding TE topology. The TE topology may or

may not be congruent to the layer 3 network topology. When such a congruent TE topology exists, there will be a one-to-one association between the one modeling element in the layer 3 topology to another element in the TE topology. When such a congruent TE topology does not exist, the association will not be one-to-one. This YANG data model allows both cases.

1.1. Terminology

The keywords "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].

The following terms are defined in [RFC7950] and are not redefined here:

- o augment
- o data model
- o data node

1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Modeling Considerations for L3 TE Topologies

2.1. Relationship Between Layer 3 Topology and TE topology

In general, layer 3 network topology model and TE topology model can be used independently. When traffic engineering is enabled on a layer 3 network topology, there will be associations between objects in layer 3 network topologies and objects in TE topologies. The properties of these relations are:

- o The associations are between objects of the same class, i.e. node to node or link to link.
- o The multiplicity of such an association is: 0..1 to 0..1. An object in a layer 3 network may have zero or one associated object in the corresponding TE network.

2.2. Relationship Modeling

YANG data type leafref is used to model the association relationship between a layer 3 network topology and a TE topology. YANG must statements are used to enforce the referenced objects are in the topologies of proper type.

2.2.1. Topology Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, the layer 3 network topology will have a reference to the corresponding TE topology. Such a reference is modeled as follows:

```
augment /nw:networks/nw:network/l3t:l3-topology-attributes:
  +--rw l3-te-topology-attributes
    +--rw network-ref?    -> /nw:networks/network/network-id
```

If the TE topology is congruent to the layer 3 network topology, the above reference can still be used to specified TE parameters defined in the TE topology model.

2.2.2. Node Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 network node may have a reference to the corresponding TE node. Such a reference is modeled as follows:

```
augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
  +--rw l3-te-node-attributes
    +--rw node-ref?      leafref
    +--rw network-ref?   -> /nw:networks/network/network-id
```

2.2.3. Link Termination Point Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 link termination point may have a reference to the corresponding TE link termination point. Such a reference is modeled as follows:

```
augment /nw:networks/nw:network/nw:node/nt:termination-point
  /l3t:l3-termination-point-attributes:
  +--rw l3-te-tp-attributes
    +--rw tp-ref?        leafref
    +--rw node-ref?      leafref
    +--rw network-ref?   -> /nw:networks/network/network-id
```

2.2.4. Link Referencing

When TE is enabled on a layer 3 network topology, if the TE topology is not congruent to the layer 3 network topology, a layer 3 link may have a reference to the corresponding TE link. Such a reference is modeled as follows:

```
augment /nw:networks/nw:network/nt:link/l3t:l3-link-attributes:
  +--rw l3-te-link-attributes
    +--rw link-ref?      leafref
    +--rw network-ref?   -> /nw:networks/network/network-id
```

2.3. Topology Type Modeling

A new topology type is defined in this document, to indicate a topology that is a layer 3 topology with TE enabled.

```
augment /nw:networks/nw:network/nw:network-types
  /l3t:l3-unicast-topology:
  +--rw l3-te!
```

3. Packet Switching Technology Extensions

3.1. Technology Specific Link Attributes

The technology agnostic TE Topology model is augmented with packet switching specific link attributes:

```
augment /nw:networks/tet:te/tet:templates/tet:link-template
  /tet:te-link-attributes
  /tet:interface-switching-capability:
  +--rw packet-switch-capable
    +--rw minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
    +--rw interface-mtu?          uint16
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes
  /tet:interface-switching-capability:
  +--rw packet-switch-capable
    +--rw minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
    +--rw interface-mtu?          uint16
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry
  /tet:interface-switching-capability:
  +--ro packet-switch-capable
    +--ro minimum-lsp-bandwidth?  rt-types:bandwidth-ieee-float32
    +--ro interface-mtu?          uint16
```

3.2. Performance Metric

[RFC7471], [RFC7810] and [RFC7823] specify TE performance metric parameters and their usage. The packet switching augmentations specified in this moducment support such a capability, which can be conditional enabled by a YANG feature "te-performance-metric".

```
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices:
    +--rw performance-metric
        +--rw measurement
            | .....
        +--rw normality
            | .....
        +--rw throttle
            | .....
```

Such an augmentation has been applied to:

- o Connectivity matrices container
- o Connectivity matrix entry
- o Local ink connectivities container
- o Local ink connectivity entry
- o TE link attributes containr in a TE link template
- o TE link attributes containr in a TE link
- o TE link attributes containr in a TE link
- o Information source entry in a TE link

4. Model Structure

4.1. Layer 3 TE Topology Module

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

```

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

```

4.2. Packet Switching TE Topology Module

This is an augmentation to base TE topology model.

```

module: ietf-te-topology-packet
  augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices:
      +--rw performance-metric
        +--rw measurement
          +--rw unidirectional-delay? uint32
          +--rw unidirectional-min-delay? uint32
          +--rw unidirectional-max-delay? uint32
          +--rw unidirectional-delay-variation? uint32
          +--rw unidirectional-packet-loss? decimal64
          +--rw unidirectional-residual-bandwidth?
            | rt-types:bandwidth-ieee-float32
          +--rw unidirectional-available-bandwidth?
            | rt-types:bandwidth-ieee-float32
          +--rw unidirectional-utilized-bandwidth?
            | rt-types:bandwidth-ieee-float32
        +--rw normality
          +--rw unidirectional-delay?
            | te-types:performance-metric-normality
          +--rw unidirectional-min-delay?

```

```

|         te-types:performance-metric-normality
+--rw unidirectional-max-delay?
|         te-types:performance-metric-normality
+--rw unidirectional-delay-variation?
|         te-types:performance-metric-normality
+--rw unidirectional-packet-loss?
|         te-types:performance-metric-normality
+--rw unidirectional-residual-bandwidth?
|         te-types:performance-metric-normality
+--rw unidirectional-available-bandwidth?
|         te-types:performance-metric-normality
+--rw unidirectional-utilized-bandwidth?
|         te-types:performance-metric-normality
+--rw throttle
+--rw unidirectional-delay-offset?                uint32
+--rw measure-interval?                          uint32
+--rw advertisement-interval?                    uint32
+--rw suppression-interval?                      uint32
+--rw threshold-out
|   +--rw unidirectional-delay?                  uint32
|   +--rw unidirectional-min-delay?              uint32
|   +--rw unidirectional-max-delay?              uint32
|   +--rw unidirectional-delay-variation?        uint32
|   +--rw unidirectional-packet-loss?            decimal64
|   +--rw unidirectional-residual-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
+--rw threshold-in
|   +--rw unidirectional-delay?                  uint32
|   +--rw unidirectional-min-delay?              uint32
|   +--rw unidirectional-max-delay?              uint32
|   +--rw unidirectional-delay-variation?        uint32
|   +--rw unidirectional-packet-loss?            decimal64
|   +--rw unidirectional-residual-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|       |   rt-types:bandwidth-ieee-float32
+--rw threshold-accelerated-advertisement
|   +--rw unidirectional-delay?                  uint32
|   +--rw unidirectional-min-delay?              uint32
|   +--rw unidirectional-max-delay?              uint32
|   +--rw unidirectional-delay-variation?        uint32
|   +--rw unidirectional-packet-loss?            decimal64

```

```

        +--rw unidirectional-residual-bandwidth?
        |   rt-types:bandwidth-ieee-float32
        +--rw unidirectional-available-bandwidth?
        |   rt-types:bandwidth-ieee-float32
        +--rw unidirectional-utilized-bandwidth?
        |   rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:te-node-attributes/tet:connectivity-matrices
    /tet:connectivity-matrix:
+--rw performance-metric
+--rw measurement
|   +--rw unidirectional-delay?                               uint32
|   +--rw unidirectional-min-delay?                           uint32
|   +--rw unidirectional-max-delay?                           uint32
|   +--rw unidirectional-delay-variation?                     uint32
|   +--rw unidirectional-packet-loss?                         decimal64
|   +--rw unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--rw normality
|   +--rw unidirectional-delay?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-min-delay?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-max-delay?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-delay-variation?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-packet-loss?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-residual-bandwidth?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-available-bandwidth?
|   |   te-types:performance-metric-normality
|   +--rw unidirectional-utilized-bandwidth?
|   |   te-types:performance-metric-normality
+--rw throttle
|   +--rw unidirectional-delay-offset?                         uint32
|   +--rw measure-interval?                                   uint32
|   +--rw advertisement-interval?                             uint32
|   +--rw suppression-interval?                               uint32
|   +--rw threshold-out
|   |   +--rw unidirectional-delay?                           uint32
|   |   +--rw unidirectional-min-delay?                       uint32
|   |   +--rw unidirectional-max-delay?                       uint32

```

```

    |   +--rw unidirectional-delay-variation?          uint32
    |   +--rw unidirectional-packet-loss?             decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
+--rw threshold-in
    |   +--rw unidirectional-delay?                   uint32
    |   +--rw unidirectional-min-delay?               uint32
    |   +--rw unidirectional-max-delay?               uint32
    |   +--rw unidirectional-delay-variation?         uint32
    |   +--rw unidirectional-packet-loss?             decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
+--rw threshold-accelerated-advertisement
    |   +--rw unidirectional-delay?                   uint32
    |   +--rw unidirectional-min-delay?               uint32
    |   +--rw unidirectional-max-delay?               uint32
    |   +--rw unidirectional-delay-variation?         uint32
    |   +--rw unidirectional-packet-loss?             decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:information-source-entry/tet:connectivity-matrices:
+--ro performance-metric
    +--ro measurement
    |   +--ro unidirectional-delay?                   uint32
    |   +--ro unidirectional-min-delay?               uint32
    |   +--ro unidirectional-max-delay?               uint32
    |   +--ro unidirectional-delay-variation?         uint32
    |   +--ro unidirectional-packet-loss?             decimal64
    |   +--ro unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
+--ro normality

```



```

|   +--ro unidirectional-delay?
|   |   te-types:performance-metric-normality
+--ro unidirectional-min-delay?
|   |   te-types:performance-metric-normality
+--ro unidirectional-max-delay?
|   |   te-types:performance-metric-normality
+--ro unidirectional-delay-variation?
|   |   te-types:performance-metric-normality
+--ro unidirectional-packet-loss?
|   |   te-types:performance-metric-normality
+--ro unidirectional-residual-bandwidth?
|   |   te-types:performance-metric-normality
+--ro unidirectional-available-bandwidth?
|   |   te-types:performance-metric-normality
+--ro unidirectional-utilized-bandwidth?
|   |   te-types:performance-metric-normality
+--ro throttle
+--ro unidirectional-delay-offset?          uint32
+--ro measure-interval?                    uint32
+--ro advertisement-interval?              uint32
+--ro suppression-interval?               uint32
+--ro threshold-out
|   +--ro unidirectional-delay?            uint32
|   +--ro unidirectional-min-delay?        uint32
|   +--ro unidirectional-max-delay?        uint32
|   +--ro unidirectional-delay-variation?  uint32
|   +--ro unidirectional-packet-loss?      decimal64
|   +--ro unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--ro threshold-in
|   +--ro unidirectional-delay?            uint32
|   +--ro unidirectional-min-delay?        uint32
|   +--ro unidirectional-max-delay?        uint32
|   +--ro unidirectional-delay-variation?  uint32
|   +--ro unidirectional-packet-loss?      decimal64
|   +--ro unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--ro threshold-accelerated-advertisement
+--ro unidirectional-delay?                uint32
+--ro unidirectional-min-delay?            uint32

```

```

        +--ro unidirectional-max-delay?          uint32
        +--ro unidirectional-delay-variation?    uint32
        +--ro unidirectional-packet-loss?        decimal64
        +--ro unidirectional-residual-bandwidth?
           | rt-types:bandwidth-ieee-float32
        +--ro unidirectional-available-bandwidth?
           | rt-types:bandwidth-ieee-float32
        +--ro unidirectional-utilized-bandwidth?
           | rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nw:node/tet:te
        /tet:information-source-entry/tet:connectivity-matrices
        /tet:connectivity-matrix:
+--ro performance-metric
  +--ro measurement
    | +--ro unidirectional-delay?          uint32
    | +--ro unidirectional-min-delay?      uint32
    | +--ro unidirectional-max-delay?      uint32
    | +--ro unidirectional-delay-variation? uint32
    | +--ro unidirectional-packet-loss?    decimal64
    | +--ro unidirectional-residual-bandwidth?
    | | rt-types:bandwidth-ieee-float32
    | +--ro unidirectional-available-bandwidth?
    | | rt-types:bandwidth-ieee-float32
    | +--ro unidirectional-utilized-bandwidth?
    | | rt-types:bandwidth-ieee-float32
  +--ro normality
    | +--ro unidirectional-delay?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-min-delay?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-max-delay?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-delay-variation?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-packet-loss?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-residual-bandwidth?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-available-bandwidth?
    | | te-types:performance-metric-normality
    | +--ro unidirectional-utilized-bandwidth?
    | | te-types:performance-metric-normality
  +--ro throttle
    +--ro unidirectional-delay-offset?      uint32
    +--ro measure-interval?                 uint32
    +--ro advertisement-interval?           uint32
    +--ro suppression-interval?             uint32
    +--ro threshold-out

```

```

    |   +--ro unidirectional-delay?                uint32
    |   +--ro unidirectional-min-delay?            uint32
    |   +--ro unidirectional-max-delay?            uint32
    |   +--ro unidirectional-delay-variation?      uint32
    |   +--ro unidirectional-packet-loss?          decimal64
    |   +--ro unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
+--ro threshold-in
    |   +--ro unidirectional-delay?                uint32
    |   +--ro unidirectional-min-delay?            uint32
    |   +--ro unidirectional-max-delay?            uint32
    |   +--ro unidirectional-delay-variation?      uint32
    |   +--ro unidirectional-packet-loss?          decimal64
    |   +--ro unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
+--ro threshold-accelerated-advertisement
    |   +--ro unidirectional-delay?                uint32
    |   +--ro unidirectional-min-delay?            uint32
    |   +--ro unidirectional-max-delay?            uint32
    |   +--ro unidirectional-delay-variation?      uint32
    |   +--ro unidirectional-packet-loss?          decimal64
    |   +--ro unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:tunnel-termination-point
    /tet:local-link-connectivities:
+--rw performance-metric
+--rw measurement
    |   +--rw unidirectional-delay?                uint32
    |   +--rw unidirectional-min-delay?            uint32
    |   +--rw unidirectional-max-delay?            uint32
    |   +--rw unidirectional-delay-variation?      uint32
    |   +--rw unidirectional-packet-loss?          decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?

```

```

|         rt-types:bandwidth-ieee-float32
|   +---rw unidirectional-utilized-bandwidth?
|         rt-types:bandwidth-ieee-float32
+---rw normality
|   +---rw unidirectional-delay?
|         te-types:performance-metric-normality
|   +---rw unidirectional-min-delay?
|         te-types:performance-metric-normality
|   +---rw unidirectional-max-delay?
|         te-types:performance-metric-normality
|   +---rw unidirectional-delay-variation?
|         te-types:performance-metric-normality
|   +---rw unidirectional-packet-loss?
|         te-types:performance-metric-normality
|   +---rw unidirectional-residual-bandwidth?
|         te-types:performance-metric-normality
|   +---rw unidirectional-available-bandwidth?
|         te-types:performance-metric-normality
|   +---rw unidirectional-utilized-bandwidth?
|         te-types:performance-metric-normality
+---rw throttle
|   +---rw unidirectional-delay-offset?          uint32
|   +---rw measure-interval?                    uint32
|   +---rw advertisement-interval?              uint32
|   +---rw suppression-interval?                uint32
|   +---rw threshold-out
|     +---rw unidirectional-delay?              uint32
|     +---rw unidirectional-min-delay?          uint32
|     +---rw unidirectional-max-delay?          uint32
|     +---rw unidirectional-delay-variation?    uint32
|     +---rw unidirectional-packet-loss?        decimal64
|     +---rw unidirectional-residual-bandwidth?
|       |         rt-types:bandwidth-ieee-float32
|     +---rw unidirectional-available-bandwidth?
|       |         rt-types:bandwidth-ieee-float32
|     +---rw unidirectional-utilized-bandwidth?
|       |         rt-types:bandwidth-ieee-float32
+---rw threshold-in
|   +---rw unidirectional-delay?              uint32
|   +---rw unidirectional-min-delay?          uint32
|   +---rw unidirectional-max-delay?          uint32
|   +---rw unidirectional-delay-variation?    uint32
|   +---rw unidirectional-packet-loss?        decimal64
|   +---rw unidirectional-residual-bandwidth?
|     |         rt-types:bandwidth-ieee-float32
|   +---rw unidirectional-available-bandwidth?
|     |         rt-types:bandwidth-ieee-float32
|   +---rw unidirectional-utilized-bandwidth?

```

```

    |           rt-types:bandwidth-ieee-float32
+--rw threshold-accelerated-advertisement
    |   +--rw unidirectional-delay?                uint32
    |   +--rw unidirectional-min-delay?            uint32
    |   +--rw unidirectional-max-delay?            uint32
    |   +--rw unidirectional-delay-variation?      uint32
    |   +--rw unidirectional-packet-loss?          decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-utilized-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nw:node/tet:te
    /tet:tunnel-termination-point
    /tet:local-link-connectivities
    /tet:local-link-connectivity:
+--rw performance-metric
    +--rw measurement
    |   +--rw unidirectional-delay?                uint32
    |   +--rw unidirectional-min-delay?            uint32
    |   +--rw unidirectional-max-delay?            uint32
    |   +--rw unidirectional-delay-variation?      uint32
    |   +--rw unidirectional-packet-loss?          decimal64
    |   +--rw unidirectional-residual-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-available-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
    |   +--rw unidirectional-utilized-bandwidth?
    |       |           rt-types:bandwidth-ieee-float32
    +--rw normality
    |   +--rw unidirectional-delay?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-min-delay?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-max-delay?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-delay-variation?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-packet-loss?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-residual-bandwidth?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-available-bandwidth?
    |       |           te-types:performance-metric-normality
    |   +--rw unidirectional-utilized-bandwidth?
    |       |           te-types:performance-metric-normality
+--rw throttle

```

```

+--rw unidirectional-delay-offset?          uint32
+--rw measure-interval?                     uint32
+--rw advertisement-interval?               uint32
+--rw suppression-interval?                uint32
+--rw threshold-out
|   +--rw unidirectional-delay?             uint32
|   +--rw unidirectional-min-delay?         uint32
|   +--rw unidirectional-max-delay?         uint32
|   +--rw unidirectional-delay-variation?   uint32
|   +--rw unidirectional-packet-loss?       decimal64
|   +--rw unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--rw threshold-in
|   +--rw unidirectional-delay?             uint32
|   +--rw unidirectional-min-delay?         uint32
|   +--rw unidirectional-max-delay?         uint32
|   +--rw unidirectional-delay-variation?   uint32
|   +--rw unidirectional-packet-loss?       decimal64
|   +--rw unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--rw threshold-accelerated-advertisement
|   +--rw unidirectional-delay?             uint32
|   +--rw unidirectional-min-delay?         uint32
|   +--rw unidirectional-max-delay?         uint32
|   +--rw unidirectional-delay-variation?   uint32
|   +--rw unidirectional-packet-loss?       decimal64
|   +--rw unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
augment /nw:networks/tet:te/tet:templates/tet:link-template
/tet:te-link-attributes:
+--rw performance-metric
+--rw measurement
|   +--rw unidirectional-delay?             uint32
|   +--rw unidirectional-min-delay?         uint32
|   +--rw unidirectional-max-delay?         uint32
|   +--rw unidirectional-delay-variation?   uint32

```

```

|   +--rw unidirectional-packet-loss?          decimal64
|   +--rw unidirectional-residual-bandwidth?
|   |       rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   |       rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   |       rt-types:bandwidth-ieee-float32
+--rw normality
|   +--rw unidirectional-delay?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-min-delay?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-max-delay?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-delay-variation?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-packet-loss?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-residual-bandwidth?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-available-bandwidth?
|   |       te-types:performance-metric-normality
|   +--rw unidirectional-utilized-bandwidth?
|   |       te-types:performance-metric-normality
+--rw throttle
|   +--rw unidirectional-delay-offset?          uint32
|   +--rw measure-interval?                    uint32
|   +--rw advertisement-interval?              uint32
|   +--rw suppression-interval?                uint32
|   +--rw threshold-out
|   |   +--rw unidirectional-delay?            uint32
|   |   +--rw unidirectional-min-delay?        uint32
|   |   +--rw unidirectional-max-delay?        uint32
|   |   +--rw unidirectional-delay-variation?  uint32
|   |   +--rw unidirectional-packet-loss?      decimal64
|   |   +--rw unidirectional-residual-bandwidth?
|   |   |       rt-types:bandwidth-ieee-float32
|   |   +--rw unidirectional-available-bandwidth?
|   |   |       rt-types:bandwidth-ieee-float32
|   |   +--rw unidirectional-utilized-bandwidth?
|   |   |       rt-types:bandwidth-ieee-float32
|   +--rw threshold-in
|   |   +--rw unidirectional-delay?            uint32
|   |   +--rw unidirectional-min-delay?        uint32
|   |   +--rw unidirectional-max-delay?        uint32
|   |   +--rw unidirectional-delay-variation?  uint32
|   |   +--rw unidirectional-packet-loss?      decimal64
|   |   +--rw unidirectional-residual-bandwidth?

```

```

    |         rt-types:bandwidth-ieee-float32
    | +--rw unidirectional-available-bandwidth?
    | |         rt-types:bandwidth-ieee-float32
    | | +--rw unidirectional-utilized-bandwidth?
    | |         rt-types:bandwidth-ieee-float32
    | +--rw threshold-accelerated-advertisement
    | +--rw unidirectional-delay?                               uint32
    | +--rw unidirectional-min-delay?                           uint32
    | +--rw unidirectional-max-delay?                           uint32
    | +--rw unidirectional-delay-variation?                     uint32
    | +--rw unidirectional-packet-loss?                         decimal64
    | +--rw unidirectional-residual-bandwidth?
    | |         rt-types:bandwidth-ieee-float32
    | | +--rw unidirectional-available-bandwidth?
    | | |         rt-types:bandwidth-ieee-float32
    | | | +--rw unidirectional-utilized-bandwidth?
    | | |         rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes:
    +--rw performance-metric
    | +--rw measurement
    | | +--rw unidirectional-delay?                               uint32
    | | +--rw unidirectional-min-delay?                           uint32
    | | +--rw unidirectional-max-delay?                           uint32
    | | +--rw unidirectional-delay-variation?                     uint32
    | | +--rw unidirectional-packet-loss?                         decimal64
    | | +--rw unidirectional-residual-bandwidth?
    | | |         rt-types:bandwidth-ieee-float32
    | | | +--rw unidirectional-available-bandwidth?
    | | | |         rt-types:bandwidth-ieee-float32
    | | | +--rw unidirectional-utilized-bandwidth?
    | | |         rt-types:bandwidth-ieee-float32
    | +--rw normality
    | | +--rw unidirectional-delay?
    | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-min-delay?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-max-delay?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-delay-variation?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-packet-loss?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-residual-bandwidth?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-available-bandwidth?
    | | | |         te-types:performance-metric-normality
    | | | +--rw unidirectional-utilized-bandwidth?

```



```

|           te-types:performance-metric-normality
+--rw throttle
|   +--rw unidirectional-delay-offset?          uint32
|   +--rw measure-interval?                     uint32
|   +--rw advertisement-interval?               uint32
|   +--rw suppression-interval?                 uint32
|   +--rw threshold-out
|       |   +--rw unidirectional-delay?          uint32
|       |   +--rw unidirectional-min-delay?      uint32
|       |   +--rw unidirectional-max-delay?      uint32
|       |   +--rw unidirectional-delay-variation? uint32
|       |   +--rw unidirectional-packet-loss?    decimal64
|       |   +--rw unidirectional-residual-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-available-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-utilized-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|   +--rw threshold-in
|       |   +--rw unidirectional-delay?          uint32
|       |   +--rw unidirectional-min-delay?      uint32
|       |   +--rw unidirectional-max-delay?      uint32
|       |   +--rw unidirectional-delay-variation? uint32
|       |   +--rw unidirectional-packet-loss?    decimal64
|       |   +--rw unidirectional-residual-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-available-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-utilized-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|   +--rw threshold-accelerated-advertisement
|       |   +--rw unidirectional-delay?          uint32
|       |   +--rw unidirectional-min-delay?      uint32
|       |   +--rw unidirectional-max-delay?      uint32
|       |   +--rw unidirectional-delay-variation? uint32
|       |   +--rw unidirectional-packet-loss?    decimal64
|       |   +--rw unidirectional-residual-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-available-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
|       |   +--rw unidirectional-utilized-bandwidth?
|       |       |   rt-types:bandwidth-ieee-float32
augment /nw:networks/nw:network/nt:link/tet:te
|   /tet:information-source-entry:
+--ro performance-metric
|   +--ro measurement
|       |   +--ro unidirectional-delay?          uint32
|       |   +--ro unidirectional-min-delay?      uint32

```

```

|   +--ro unidirectional-max-delay?          uint32
|   +--ro unidirectional-delay-variation?    uint32
|   +--ro unidirectional-packet-loss?        decimal64
|   +--ro unidirectional-residual-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-available-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
|   +--ro unidirectional-utilized-bandwidth?
|   |   rt-types:bandwidth-ieee-float32
+--ro normality
|   +--ro unidirectional-delay?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-min-delay?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-max-delay?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-delay-variation?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-packet-loss?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-residual-bandwidth?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-available-bandwidth?
|   |   te-types:performance-metric-normality
|   +--ro unidirectional-utilized-bandwidth?
|   |   te-types:performance-metric-normality
+--ro throttle
|   +--ro unidirectional-delay-offset?        uint32
|   +--ro measure-interval?                  uint32
|   +--ro advertisement-interval?            uint32
|   +--ro suppression-interval?              uint32
|   +--ro threshold-out
|   |   +--ro unidirectional-delay?          uint32
|   |   +--ro unidirectional-min-delay?      uint32
|   |   +--ro unidirectional-max-delay?      uint32
|   |   +--ro unidirectional-delay-variation? uint32
|   |   +--ro unidirectional-packet-loss?    decimal64
|   |   +--ro unidirectional-residual-bandwidth?
|   |   |   rt-types:bandwidth-ieee-float32
|   |   +--ro unidirectional-available-bandwidth?
|   |   |   rt-types:bandwidth-ieee-float32
|   |   +--ro unidirectional-utilized-bandwidth?
|   |   |   rt-types:bandwidth-ieee-float32
|   +--ro threshold-in
|   |   +--ro unidirectional-delay?          uint32
|   |   +--ro unidirectional-min-delay?      uint32
|   |   +--ro unidirectional-max-delay?      uint32
|   |   +--ro unidirectional-delay-variation? uint32

```

```

    |   +--ro unidirectional-packet-loss?          decimal64
    |   +--ro unidirectional-residual-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-available-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    |   +--ro unidirectional-utilized-bandwidth?
    |   |       rt-types:bandwidth-ieee-float32
    +--ro threshold-accelerated-advertisement
    +--ro unidirectional-delay?                    uint32
    +--ro unidirectional-min-delay?                 uint32
    +--ro unidirectional-max-delay?                 uint32
    +--ro unidirectional-delay-variation?           uint32
    +--ro unidirectional-packet-loss?              decimal64
    +--ro unidirectional-residual-bandwidth?
    |       rt-types:bandwidth-ieee-float32
    +--ro unidirectional-available-bandwidth?
    |       rt-types:bandwidth-ieee-float32
    +--ro unidirectional-utilized-bandwidth?
    |       rt-types:bandwidth-ieee-float32
augment /nw:networks/tet:te/tet:templates/tet:link-template
  /tet:te-link-attributes
  /tet:interface-switching-capability:
    +--rw packet-switch-capable
    +--rw minimum-lsp-bandwidth?   rt-types:bandwidth-ieee-float32
    +--rw interface-mtu?           uint16
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:te-link-attributes
  /tet:interface-switching-capability:
    +--rw packet-switch-capable
    +--rw minimum-lsp-bandwidth?   rt-types:bandwidth-ieee-float32
    +--rw interface-mtu?           uint16
augment /nw:networks/nw:network/nt:link/tet:te
  /tet:information-source-entry
  /tet:interface-switching-capability:
    +--ro packet-switch-capable
    +--ro minimum-lsp-bandwidth?   rt-types:bandwidth-ieee-float32
    +--ro interface-mtu?           uint16

```

5. YANG Modules

5.1. Layer 3 TE Topology Module

This module references [RFC8345], [RFC8346], and [I-D.ietf-teas-yang-te-topo].

<CODE BEGINS> file "ietf-l3-te-topology@2018-06-22.yang"

```
module ietf-l3-te-topology {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l3-te-topology";
  prefix "l3tet";

  import ietf-network {
    prefix "nw";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-network-topology {
    prefix "nt";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-l3-unicast-topology {
    prefix "l3t";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }
  import ietf-te-topology {
    prefix "tet";
    reference
      "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic
       Engineering (TE) Topologies";
  }

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

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

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

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

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

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

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

     Editor:    Oscar Gonzalez De Dios
```

<mailto:oscar.gonzalezdedios@telefonica.com>;

description

"YANG data model for representing and manipulating Layer 3 TE Topologies.

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

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions 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 2018-06-22 {

description "Initial revision";
reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";

}

grouping l3-te-topology-type {

description
"Identifies the L3 TE topology type."
container l3-te {
presence "indicates L3 TE Topology";
description
"Its presence identifies the L3 TE topology type."
}

}

augment "/nw:networks/nw:network/nw:network-types/"

+ "l3t:l3-unicast-topology" {
description
"Defines the L3 TE topology type."
uses l3-te-topology-type;

}

augment "/nw:networks/nw:network/l3t:l3-topology-attributes" {

when "../nw:network-types/l3t:l3-unicast-topology/l3tet:l3-te" {
description "Augment only for L3 TE topology";
}
description "Augment topology configuration";
uses l3-te-topology-attributes;

```
}

augment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes" {
  when "../nw:network-types/l3t:l3-unicast-topology/"
    + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment node configuration";
  uses l3-te-node-attributes;
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/"
  + "l3t:l3-termination-point-attributes" {
  when "../nw:network-types/l3t:l3-unicast-topology/"
    + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment termination point configuration";
  uses l3-te-tp-attributes;
}

augment "/nw:networks/nw:network/nt:link/l3t:l3-link-attributes" {
  when "../nw:network-types/l3t:l3-unicast-topology/"
    + "l3tet:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment link configuration";
  uses l3-te-link-attributes;
}

grouping l3-te-topology-attributes {
  description "L3 TE topology scope attributes";
  container l3-te-topology-attributes {
    must "/nw:networks/nw:network"
      + "[nw:network-id = current()/network-ref]/nw:network-types/"
      + "tet:te-topology" {
      error-message
        "The referenced network must be a TE topology.";
      description
        "The referenced network must be a TE topology.";
    }
    description "Containing TE topology references";
    uses nw:network-ref;
  } // l3-te-topology-attributes
} // l3-te-topology-attributes

grouping l3-te-node-attributes {
  description "L3 TE node scope attributes";
```

```
    container l3-te-node-attributes {
      must "/nw:networks/nw:network"
      + "[nw:network-id = current()/network-ref]/nw:network-types/"
      + "tet:te-topology" {
        error-message
          "The referenced network must be a TE topology.";
        description
          "The referenced network must be a TE topology.";
      }
      description "Containing TE node references";
      uses nw:node-ref;
    } // l3-te
  } // l3-te-node-attributes

  grouping l3-te-tp-attributes {
    description "L3 TE termination point scope attributes";
    container l3-te-tp-attributes {
      must "/nw:networks/nw:network"
      + "[nw:network-id = current()/network-ref]/nw:network-types/"
      + "tet:te-topology" {
        error-message
          "The referenced network must be a TE topology.";
        description
          "The referenced network must be a TE topology.";
      }
      description "Containing TE termination point references";
      uses nt:tp-ref;
    } // l3-te
  } // l3-te-tp-attributes

  grouping l3-te-link-attributes {
    description "L3 TE link scope attributes";
    container l3-te-link-attributes {
      must "/nw:networks/nw:network"
      + "[nw:network-id = current()/network-ref]/nw:network-types/"
      + "tet:te-topology" {
        error-message
          "The referenced network must be a TE topology.";
        description
          "The referenced network must be a TE topology.";
      }
      description "Containing TE link references";
      uses nt:link-ref;
    }
  } // l3-te-link-attributes
}
<CODE ENDS>
```

5.2. Packet Switching TE Topology Module

This module references [RFC7471], [RFC7810], [RFC7823], [RFC8294], [RFC8345], [RFC8346]. [I-D.ietf-teas-yang-te], and [I-D.ietf-teas-yang-te-topo].

```
<CODE BEGINS> file "ietf-te-topology-packet@2018-06-22.yang"
module ietf-te-topology-packet {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology-packet";

  prefix "tet-pkt";

  import ietf-network {
    prefix "nw";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-network-topology {
    prefix "nt";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-te-topology {
    prefix "tet";
    reference
      "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic
      Engineering (TE) Topologies";
  }

  import ietf-te-types {
    prefix "te-types";
    reference
      "I-D.ietf-teas-yang-te: A YANG Data Model for Traffic
      Engineering Tunnels and Interfaces";
  }

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


contact

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

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

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

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

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

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

Editor: Oscar Gonzalez De Dios
<<mailto:oscar.gonzalezdedios@telefonica.com>>" ;

description

"YANG data model for representing and manipulating PSC (Packet Switching) TE Topologies.

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

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions 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 2018-06-22 {  
  description "Initial revision";  
  reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";  
}
```

```
/*  
 * Features  
 */
```

```
feature te-performance-metric {
  description
    "This feature indicates that the system supports
    TE performance metric.";
  reference
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
    RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
    RFC7823: Performance-Based Path Selection for Explicitly
    Routed Label Switched Paths (LSPs) Using TE Metric
    Extensions";
}

/*
 * Groupings
 */
grouping packet-switch-capable-container {
  description
    "The container of packet switch capable attributes.";
  container packet-switch-capable {
    description
      "Interface has packet-switching capabilities.";
    leaf minimum-lsp-bandwidth {
      type rt-types:bandwidth-ieee-float32;
      description
        "Minimum LSP Bandwidth. Units in bytes per second";
    }
    leaf interface-mtu {
      type uint16;
      description
        "Interface MTU.";
    }
  }
}

/*
 * Augmentations
 */
/* Augmentations to connectivity-matrix */
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices" {
  description
    "Parameters for PSC TE topology.";
  uses te-types:performance-metric-container {
    if-feature te-performance-metric;
  }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
```

```
        + "tet:te-node-attributes/tet:connectivity-matrices/"
        + "tet:connectivity-matrix" {
    description
        "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:information-source-entry/tet:connectivity-matrices" {
    description
        "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:information-source-entry/tet:connectivity-matrices/"
    + "tet:connectivity-matrix" {
    description
        "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}

/* Augmentations to tunnel-termination-point */
augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:tunnel-termination-point/"
    + "tet:local-link-connectivities" {
    description
        "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:tunnel-termination-point/"
    + "tet:local-link-connectivities/"
    + "tet:local-link-connectivity" {
    description
        "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}
```

```
    }

    /* Augmentations to te-link-attributes */
    augment "/nw:networks/tet:te/tet:templates/"
      + "tet:link-template/tet:te-link-attributes" {
      when "tet:interface-switching-capability "
        + "[tet:switching-capability = 'te-types:switching-psc1']" {
        description "Valid only for PSC";
      }
      description
        "Parameters for PSC TE topology.";
      uses te-types:performance-metric-container {
        if-feature te-performance-metric;
      }
    }

    augment "/nw:networks/nw:network/nt:link/tet:te/"
      + "tet:te-link-attributes" {
      when "tet:interface-switching-capability "
        + "[tet:switching-capability = 'te-types:switching-psc1']" {
        description "Valid only for PSC";
      }
      description
        "Parameters for PSC TE topology.";
      uses te-types:performance-metric-container {
        if-feature te-performance-metric;
      }
    }

    augment "/nw:networks/nw:network/nt:link/tet:te/"
      + "tet:information-source-entry" {
      when "tet:interface-switching-capability "
        + "[tet:switching-capability = 'te-types:switching-psc1']" {
        description "Valid only for PSC";
      }
      description
        "Parameters for PSC TE topology.";
      uses te-types:performance-metric-container {
        if-feature te-performance-metric;
      }
    }

    /* Augmentations to interface-switching-capability */
    augment "/nw:networks/tet:te/tet:templates/"
      + "tet:link-template/tet:te-link-attributes/"
      + "tet:interface-switching-capability" {
      when "tet:switching-capability = 'te-types:switching-psc1' " {
        description "Valid only for PSC";
      }
    }
```

```

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

  augment "/nw:networks/nw:network/nt:link/tet:te/"
    + "tet:te-link-attributes/"
    + "tet:interface-switching-capability" {
    when "tet:switching-capability = 'te-types:switching-psc1' " {
      description "Valid only for PSC";
    }
    description
      "Parameters for PSC TE topology.";
    uses packet-switch-capable-container;
  }

  augment "/nw:networks/nw:network/nt:link/tet:te/"
    + "tet:information-source-entry/"
    + "tet:interface-switching-capability" {
    when "tet:switching-capability = 'te-types:switching-psc1' " {
      description "Valid only for PSC";
    }
    description
      "Parameters for PSC TE topology.";
    uses packet-switch-capable-container;
  }
}
<CODE ENDS>

```

6. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

```

-----
URI: urn:ietf:params:xml:ns:yang:ietf-l3-te-topology
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
-----

```

URI: urn:ietf:params:xml:ns:yang:ietf-l3-te-topology-state
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

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

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

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

name: ietf-l3-te-topology
namespace: urn:ietf:params:xml:ns:yang:ietf-l3-te-topology
prefix: l3te
reference: RFC XXXX

name: ietf-l3-te-topology-state
namespace: urn:ietf:params:xml:ns:yang:ietf-l3-te-topology-state
prefix: l3te-s
reference: RFC XXXX

name: ietf-te-topology-packet
namespace: urn:ietf:params:xml:ns:yang:ietf-te-topology-packet
prefix: tet-pkt
reference: RFC XXXX

name: ietf-te-topology-packet-state
namespace: urn:ietf:params:xml:ns:yang:ietf-te-topology-packet-state
prefix: tet-pkt-s
reference: RFC XXXX

7. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

`/nw:networks/nw:network/nw:network-types/l3t:l3-unicast-topology/l3-te`

This subtree specifies the layer 3 TE topology type. Modifying the configurations can make layer 3 TE topology type invalid and cause interruption to all layer 3 TE networks.

`/nw:networks/nw:network/l3t:l3-topology-attributes/l3-te-topology-attributes`

This subtree specifies the topology-wide configurations, including the reference to a TE topology from a layer 3 network topology. Modifying the configurations here can cause traffic disabled or rerouted in this topology and the connected topologies.

`/nw:networks/nw:network/nw:node/l3t:l3-node-attributes/l3-te-node-attributes`

This subtree specifies the configurations of layer 3 TE nodes. Modifying the configurations in this subtree can change the relationship between a TE node and a layer 3 node, causing traffic disabled or rerouted in the specified nodes and the related layer 3 topologies.

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

This subtree specifies the configurations of layer 3 TE link termination points. Modifying the configurations in this subtree

can change the relationship between a TE link termination point and a layer 3 link termination point, causing traffic disabled or rerouted on the related layer 3 links and the related layer 3 topologies.

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

This subtree specifies the configurations of layer 3 TE links. Modifying the configurations in this subtree can change the relationship between a TE link and a layer 3 link, causing traffic disabled or rerouted on the specified layer 3 link and the related layer 3 topologies.

performance-metric containers

The container "performance-metric" is augmented to multiple locations of the base TE topology model, as specified in Section 3.2. Modifying the configuration in such a container can change the behaviours of performance metric monitoring, causing traffic disabled or rerouted on the related layer 3 links, nodes, or topologies.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

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

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

/nw:networks/nw:network/l3t:l3-topology-attributes/l3-te-topology-attributes

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

/nw:networks/nw:network/nw:node/l3t:l3-node-attributes/l3-te-node-attributes

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

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

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

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

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

performance-metric containers

The container "performance-metric" is augmented to multiple locations of the base TE topology model, as specified in Section 3.2. Unauthorized access to this subtree can disclose the operational state information of performance metric monitoring.

8. References

8.1. Normative References

- [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>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [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>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [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>>.
- [RFC7810] Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 7810, DOI 10.17487/RFC7810, May 2016, <<https://www.rfc-editor.org/info/rfc7810>>.

- [RFC7823] Atlas, A., Drake, J., Giacalone, S., and S. Previdi, "Performance-Based Path Selection for Explicitly Routed Label Switched Paths (LSPs) Using TE Metric Extensions", RFC 7823, DOI 10.17487/RFC7823, May 2016, <<https://www.rfc-editor.org/info/rfc7823>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8346] Clemm, A., Medved, J., Varga, R., Liu, X., Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model for Layer 3 Topologies", RFC 8346, DOI 10.17487/RFC8346, March 2018, <<https://www.rfc-editor.org/info/rfc8346>>.
- [I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-15 (work in progress), June 2018.
- [I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-18 (work in progress), June 2018.

8.2. Informative References

- [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>>.

Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG modules `ietf-l3-te-topology` and `ietf-te-topology-packet` defined in this document are designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion modules, `ietf-l3-te-topology-state` and `ietf-te-topology-packet-state`, are defined as state models, which mirror the modules `ietf-l3-te-topology` and `ietf-te-topology-packet` defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion modules, `ietf-l3-te-topology-state` and `ietf-te-topology-packet-state`, are redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion modules mirrors that of the cooresponding NMDA models, the YANG trees of the companion modules are not depicted separately.

A.1. Layer 3 TE Topology State Module

This module references [RFC8345], and [RFC8346].

```
<CODE BEGINS> file "ietf-l3-te-topology-state@2018-06-22.yang"
module ietf-l3-te-topology-state {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l3-te-topology-state";
  prefix "l3tet-s";

  import ietf-l3-te-topology {
    prefix "l3tet";
  }
  import ietf-network-state {
    prefix "nw-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-network-topology-state {
    prefix "nt-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }
}
```

organization

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

contact

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

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

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

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

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

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

Editor: Oscar Gonzalez De Dios
<<mailto:oscar.gonzalezdedios@telefonica.com>>;

description

"YANG data model for representing operational state information
of Layer 3 TE Topologies, when NMDA is not supported.

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

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

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

revision 2018-06-22 {

description "Initial revision";
reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";
}

```
augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
+ "l3t-s:l3-unicast-topology" {
  description
    "Defines the L3 TE topology type.";
  uses l3tet:l3-te-topology-type;
}

augment "/nw-s:networks/nw-s:network/"
+ "l3t-s:l3-topology-attributes" {
  when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
  + "l3tet-s:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment topology configuration";
  uses l3tet:l3-te-topology-attributes;
}

augment "/nw-s:networks/nw-s:network/nw-s:node/"
+ "l3t-s:l3-node-attributes" {
  when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
  + "l3tet-s:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment node configuration";
  uses l3tet:l3-te-node-attributes;
}

augment "/nw-s:networks/nw-s:network/nw-s:node/"
+ "nt-s:termination-point/"
+ "l3t-s:l3-termination-point-attributes" {
  when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
  + "l3tet-s:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment termination point configuration";
  uses l3tet:l3-te-tp-attributes;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/"
+ "l3t-s:l3-link-attributes" {
  when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
  + "l3tet-s:l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment link configuration";
  uses l3tet:l3-te-link-attributes;
}
```

<CODE ENDS>

A.2. Packet Switching TE Topology State Module

```
<CODE BEGINS> file "ietf-te-topology-packet-state@2018-06-22.yang"
module ietf-te-topology-packet-state {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-te-topology-packet-state";

  prefix "tet-pkt-s";

  import ietf-te-topology-packet {
    prefix "tet-pkt";
  }

  import ietf-network-state {
    prefix "nw-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-network-topology-state {
    prefix "nt-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-te-topology-state {
    prefix "tet-s";
    reference
      "I-D.ietf-teas-yang-te-topo: YANG Data Model for Traffic
       Engineering (TE) Topologies";
  }

  import ietf-te-types {
    prefix "te-types";
    reference
      "I-D.ietf-teas-yang-te: A YANG Data Model for Traffic
       Engineering Tunnels and Interfaces";
  }

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

  contact
```

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

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

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

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

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

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

Editor: Oscar Gonzalez De Dios
<<mailto:oscar.gonzalezdedios@telefonica.com>>";

description

"YANG data model for representing operational state information of PSC (Packet Switching) TE Topologies, when NMDA is not supported.

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

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

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

```
revision 2018-06-22 {  
  description "Initial revision";  
  reference "RFC XXXX: YANG Data Model for Layer 3 TE Topologies";  
}  
  
/*  
 * Augmentations  
 */  
/* Augmentations to connectivity-matrix */
```



```
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:te-node-attributes/tet-s:connectivity-matrices" {
  description
    "Parameters for PSC (Packet Switching) TE topology.";
  uses te-types:performance-metric-container {
    if-feature tet-pkt:te-performance-metric;
  }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:te-node-attributes/tet-s:connectivity-matrices/"
+ "tet-s:connectivity-matrix" {
  description
    "Parameters for PSC TE topology.";
  uses te-types:performance-metric-container {
    if-feature tet-pkt:te-performance-metric;
  }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:information-source-entry/"
+ "tet-s:connectivity-matrices" {
  description
    "Parameters for PSC TE topology.";
  uses te-types:performance-metric-container {
    if-feature tet-pkt:te-performance-metric;
  }
}

augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:information-source-entry/"
+ "tet-s:connectivity-matrices/"
+ "tet-s:connectivity-matrix" {
  description
    "Parameters for PSC TE topology.";
  uses te-types:performance-metric-container {
    if-feature tet-pkt:te-performance-metric;
  }
}

/* Augmentations to tunnel-termination-point */
augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
+ "tet-s:tunnel-termination-point/"
+ "tet-s:local-link-connectivities" {
  description
    "Parameters for PSC TE topology.";
  uses te-types:performance-metric-container {
    if-feature tet-pkt:te-performance-metric;
  }
}
```

```

    }
  }

  augment "/nw-s:networks/nw-s:network/nw-s:node/tet-s:te/"
    + "tet-s:tunnel-termination-point/"
    + "tet-s:local-link-connectivities/"
    + "tet-s:local-link-connectivity" {
    description
      "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
      if-feature tet-pkt:te-performance-metric;
    }
  }

  /* Augmentations to te-link-attributes */
  augment "/nw-s:networks/tet-s:te/tet-s:templates/"
    + "tet-s:link-template/tet-s:te-link-attributes" {
    when "tet-s:interface-switching-capability "
      + "[tet-s:switching-capability = 'te-types:switching-psc1']" {
      description "Valid only for PSC";
    }
    description
      "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
      if-feature tet-pkt:te-performance-metric;
    }
  }

  augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
    + "tet-s:te-link-attributes" {
    when "tet-s:interface-switching-capability "
      + "[tet-s:switching-capability = 'te-types:switching-psc1']" {
      description "Valid only for PSC";
    }
    description
      "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
      if-feature tet-pkt:te-performance-metric;
    }
  }

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

```

```

    "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature tet-pkt:te-performance-metric;
    }
}

/* Augmentations to interface-switching-capability */
augment "/nw-s:networks/tet-s:te/tet-s:templates/"
    + "tet-s:link-template/tet-s:te-link-attributes/"
    + "tet-s:interface-switching-capability" {
    when "tet-s:switching-capability = 'te-types:switching-psc1' " {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology.";
    uses tet-pkt:packet-switch-capable-container;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
    + "tet-s:te-link-attributes/"
    + "tet-s:interface-switching-capability" {
    when "tet-s:switching-capability = 'te-types:switching-psc1' " {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology.";
    uses tet-pkt:packet-switch-capable-container;
}

augment "/nw-s:networks/nw-s:network/nt-s:link/tet-s:te/"
    + "tet-s:information-source-entry/"
    + "tet-s:interface-switching-capability" {
    when "tet-s:switching-capability = 'te-types:switching-psc1' " {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology.";
    uses tet-pkt:packet-switch-capable-container;
}
}
<CODE ENDS>

```

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
Cisco Systems Inc

EMail: tsaad@cisco.com

Himanshu Shah
Ciena

EMail: hshah@ciena.com

Oscar Gonzalez de Dios
Telefonica

EMail: oscar.gonzalezdedios@telefonica.com

TEAS Working Group
Internet Draft
Intended status: Standard Track
Expires: December 2018

Italo Busi (Ed.)
Huawei
Sergio Belotti (Ed.)
Nokia
Victor Lopez
Oscar Gonzalez de Dios
Telefonica
Anurag Sharma
Google
Yan Shi
China Unicom
Ricard Vilalta
CTTC
Karthik Sethuraman
NEC

June 29, 2018

Yang model for requesting Path Computation
draft-ietf-teas-yang-path-computation-02.txt

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on December 29, 2016.

Copyright Notice

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

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

Abstract

There are scenarios, typically in a hierarchical SDN context, where the topology information provided by a TE network provider may not be sufficient for its client to perform end-to-end path computation. In these cases the client would need to request the provider to calculate some (partial) feasible paths.

This document defines a YANG data model for a stateless RPC to request path computation. This model complements the stateful solution defined in [TE-TUNNEL].

Moreover this document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

Table of Contents

1. Introduction.....	3
1.1. Terminology.....	5
2. Use Cases.....	5
2.1. Packet/Optical Integration.....	5
2.2. Multi-domain TE Networks.....	10
2.3. Data center interconnections.....	12

3. Motivations.....	14
3.1. Motivation for a YANG Model.....	14
3.1.1. Benefits of common data models.....	14
3.1.2. Benefits of a single interface.....	15
3.1.3. Extensibility.....	15
3.2. Interactions with TE Topology.....	16
3.2.1. TE Topology Aggregation.....	17
3.2.2. TE Topology Abstraction.....	20
3.2.3. Complementary use of TE topology and path computation.....	21
3.3. Stateless and Stateful Path Computation.....	24
4. Path Computation and Optimization for multiple paths.....	25
5. YANG Model for requesting Path Computation.....	26
5.1. Synchronization of multiple path computation requests....	26
5.2. Returned metric values.....	28
6. YANG model for stateless TE path computation.....	29
6.1. YANG Tree.....	29
6.2. YANG Module.....	38
7. Security Considerations.....	47
8. IANA Considerations.....	48
9. References.....	48
9.1. Normative References.....	48
9.2. Informative References.....	49
10. Acknowledgments.....	50
Appendix A. Examples of dimensioning the "detailed connectivity matrix".....	51

1. Introduction

There are scenarios, typically in a hierarchical SDN context, where the topology information provided by a TE network provider may not be sufficient for its client to perform end-to-end path computation. In these cases the client would need to request the provider to calculate some (partial) feasible paths, complementing his topology knowledge, to make his end-to-end path computation feasible.

This type of scenarios can be applied to different interfaces in different reference architectures:

- o ABNO control interface [RFC7491], in which an Application Service Coordinator can request ABNO controller to take in charge path calculation (see Figure 1 in [RFC7491]).

- o ACTN [ACTN-frame], where a controller hierarchy is defined, the need for path computation arises on both interfaces CMI (interface between Customer Network Controller (CNC) and Multi Domain Service Coordinator (MDSC)) and/or MPI (interface between MSDC-PNC). [ACTN-Info] describes an information model for the Path Computation request.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).

Path Computation Elements, Controllers and Orchestrators perform their operations based on Traffic Engineering Databases (TED). Such TEDs can be described, in a technology agnostic way, with the YANG Data Model for TE Topologies [TE-TOPO]. Furthermore, the technology specific details of the TED are modeled in the augmented TE topology models (e.g. [OTN-TOPO] for OTN ODU technologies).

The availability of such topology models allows providing the TED using YANG-based protocols (e.g., NETCONF or RESTCONF). Furthermore, it enables a PCE/Controller performing the necessary abstractions or modifications and offering this customized topology to another PCE/Controller or high level orchestrator.

Note: This document assumes that the client of the YANG data model defined in this document may not implement a "PCE" functionality, as defined in [RFC4655].

The tunnels that can be provided over the networks described with the topology models can be also set-up, deleted and modified via YANG-based protocols (e.g., NETCONF or RESTCONF) using the TE-Tunnel Yang model [TE-TUNNEL].

This document proposes a YANG model for a path computation request defined as a stateless RPC, which complements the stateful solution defined in [TE-TUNNEL].

Moreover, this document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

1.1. Terminology

TED: The traffic engineering database is a collection of all TE information about all TE nodes and TE links in a given network.

PCE: A Path Computation Element (PCE) is an entity that is capable of computing a network path or route based on a network graph, and of applying computational constraints during the computation. The PCE entity is an application that can be located within a network node or component, on an out-of-network server, etc. For example, a PCE would be able to compute the path of a TE LSP by operating on the TED and considering bandwidth and other constraints applicable to the TE LSP service request. [RFC4655]

2. Use Cases

This section presents different use cases, where a client needs to request underlying SDN controllers for path computation.

The presented use cases have been grouped, depending on the different underlying topologies: a) Packet-Optical integration; b) Multi-domain Traffic Engineered (TE) Networks; and c) Data center interconnections.

2.1. Packet/Optical Integration

In this use case, an Optical network is used to provide connectivity to some nodes of a Packet network (see Figure 1).

A possible example could be the case where an Optical network provides connectivity to same IP routers of an IP network.

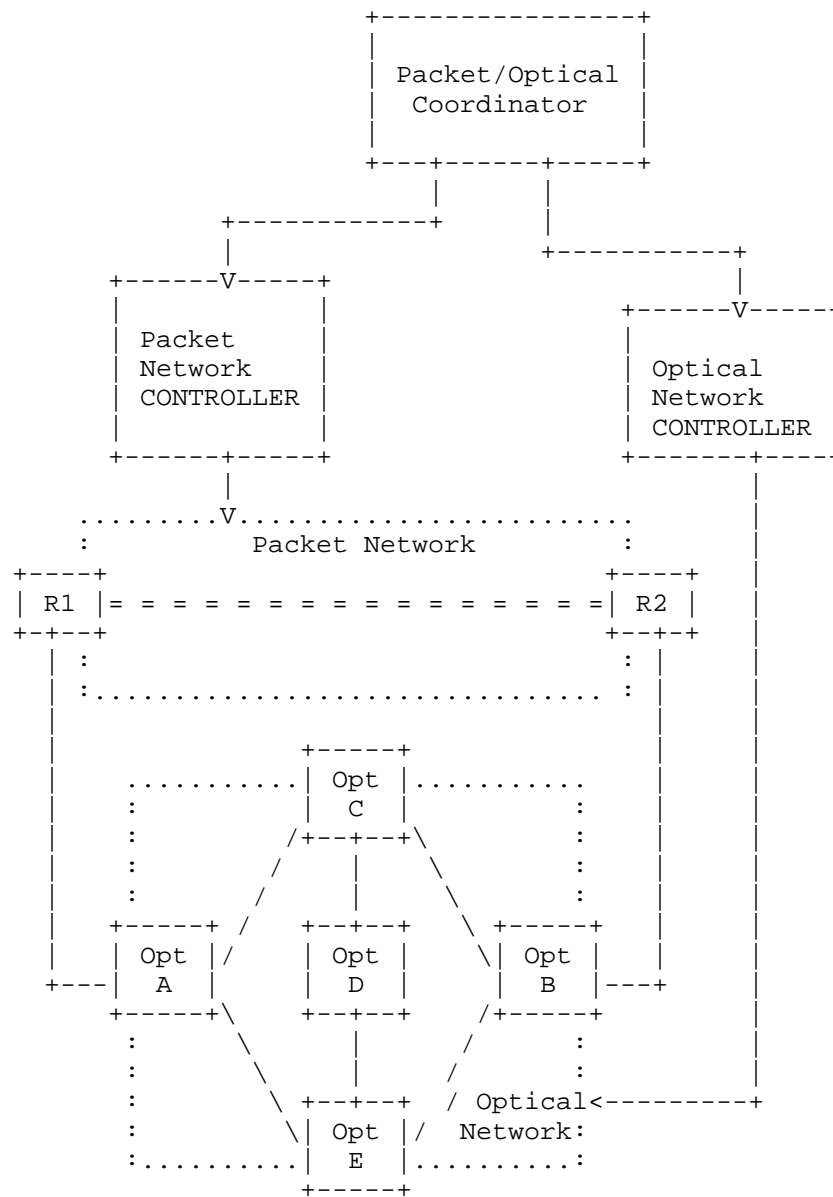


Figure 1 - Packet/Optical Integration Use Case

Figure 1 as well as Figure 2 below only show a partial view of the packet network connectivity, before additional packet connectivity is provided by the Optical network.

It is assumed that the Optical network controller provides to the packet/optical coordinator an abstracted view of the Optical network. A possible abstraction could be to represent the whole optical network as one "virtual node" with "virtual ports" connected to the access links, as shown in Figure 2.

It is also assumed that Packet network controller can provide the packet/optical coordinator the information it needs to setup connectivity between packet nodes through the Optical network (e.g., the access links).

The path computation request helps the coordinator to know the real connections that can be provided by the optical network.

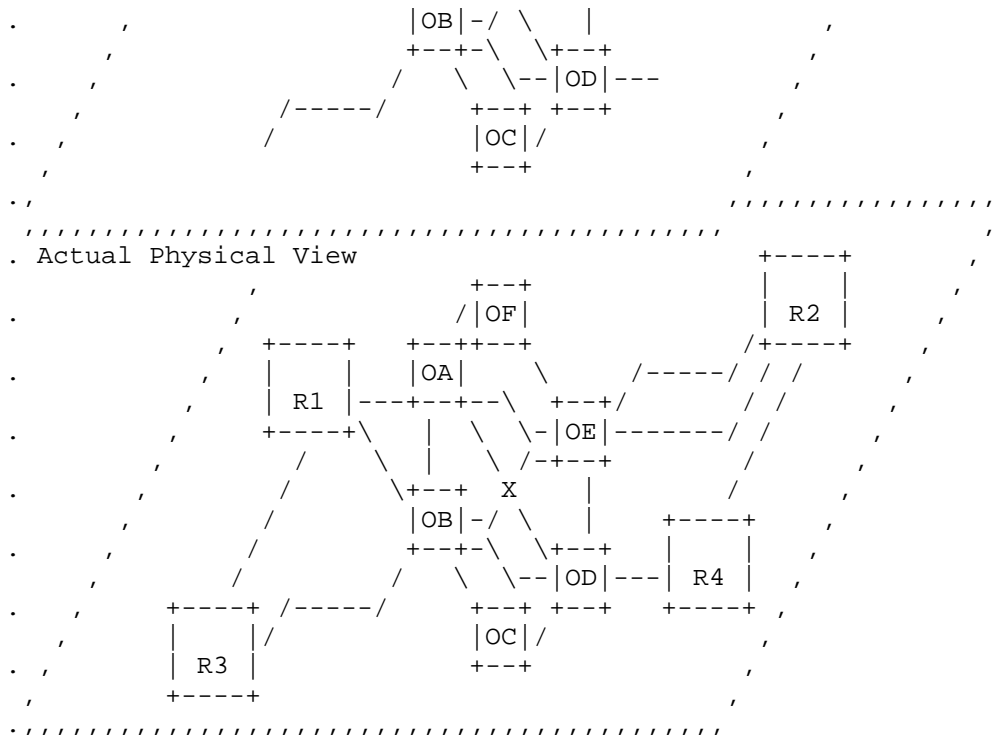


Figure 2 - Packet and Optical Topology Abstractions

In this use case, the coordinator needs to setup an optimal underlying path for an IP link between R1 and R2.

As depicted in Figure 2, the coordinator has only an "abstracted view" of the physical network, and it does not know the feasibility or the cost of the possible optical paths (e.g., VP1-VP4 and VP2-VP5), which depend from the current status of the physical resources within the optical network and on vendor-specific optical attributes.

The coordinator can request the underlying Optical domain controller to compute a set of potential optimal paths, taking into account optical constraints. Then, based on its own constraints, policy and knowledge (e.g. cost of the access links), it can choose which one of these potential paths to use to setup the optimal end-to-end path crossing optical network.

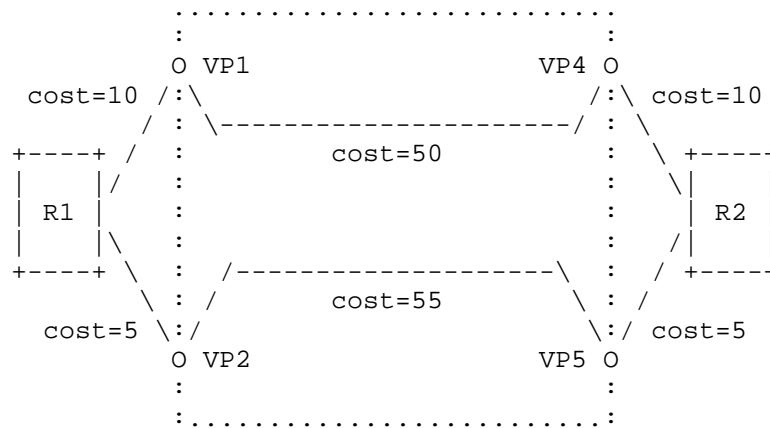


Figure 3 - Packet/Optical Path Computation Example

For example, in Figure 3, the Coordinator can request the Optical network controller to compute the paths between VP1-VP4 and VP2-VP5 and then decide to setup the optimal end-to-end path using the VP2-VP5 Optical path even this is not the optimal path from the Optical domain perspective.

Considering the dynamicity of the connectivity constraints of an Optical domain, it is possible that a path computed by the Optical network controller when requested by the Coordinator is no longer valid/available when the Coordinator requests it to be setup up. This is further discussed in section 3.3.

2.2. Multi-domain TE Networks

In this use case there are two TE domains which are interconnected together by multiple inter-domains links.

A possible example could be a multi-domain optical network.

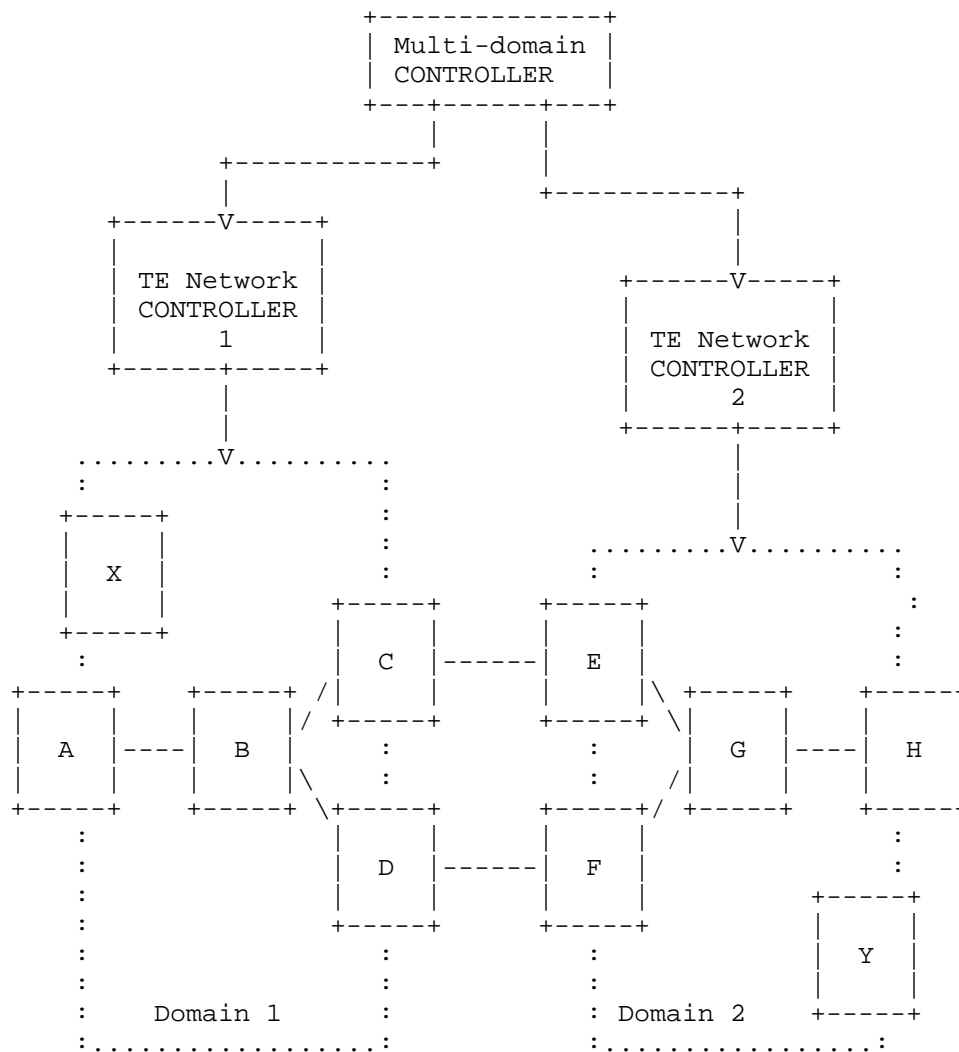


Figure 4 - Multi-domain multi-link interconnection

In order to setup an end-to-end multi-domain TE path (e.g., between nodes A and H), the multi-domain controller needs to know the feasibility or the cost of the possible TE paths within the two TE domains, which depend from the current status of the physical resources within each TE network. This is more challenging in case of optical networks because the optimal paths depend also on vendor-

specific optical attributes (which may be different in the two domains if they are provided by different vendors).

In order to setup a multi-domain TE path (e.g., between nodes A and H), the multi-domain controller can request the TE domain controllers to compute a set of intra-domain optimal paths and take decisions based on the information received. For example:

- o The Orchestrator asks TE domain controllers to provide set of paths between A-C, A-D, E-H and F-H
- o TE domain controllers return a set of feasible paths with the associated costs: the path A-C is not part of this set (in optical networks, it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller)
- o The multi-domain controller will select the path A-D-F-H since it is the only feasible multi-domain path and then request the TE domain controllers to setup the A-D and F-H intra-domain paths
- o If there are multiple feasible paths, the multi-domain controller can select the optimal path knowing the cost of the intra-domain paths (provided by the TE domain controllers) and the cost of the inter-domain links (known by the multi-domain controller)

This approach may have some scalability issues when the number of TE domains is quite big (e.g. 20).

In this case, it would be worthwhile using the abstract TE topology information provided by the domain controllers to limit the number of potential optimal end-to-end paths and then request path computation to fewer domain controllers in order to decide what the optimal path within this limited set is.

For more details, see section 3.2.3.

2.3. Data center interconnections

In these use case, there is a TE domain which is used to provide connectivity between data centers which are connected with the TE domain using access links.

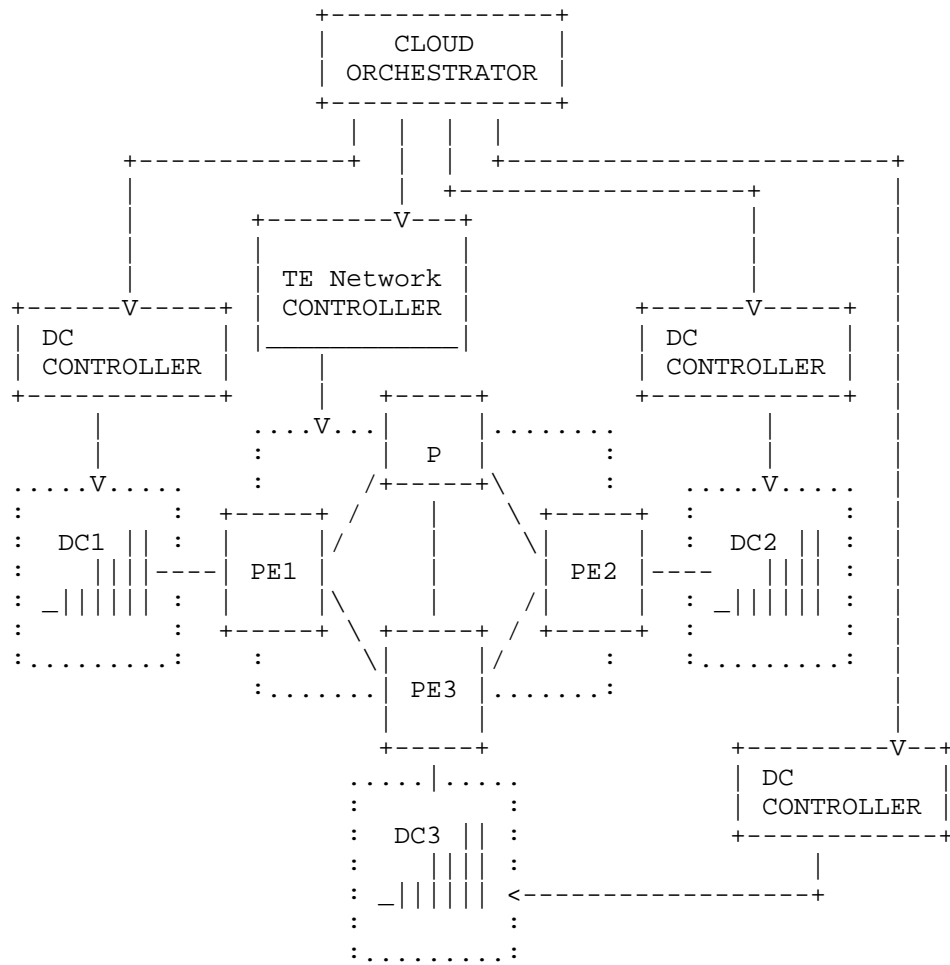


Figure 5 - Data Center Interconnection Use Case

In this use case, there is need to transfer data from Data Center 1 (DC1) to either DC2 or DC3 (e.g. workload migration).

The optimal decision depends both on the cost of the TE path (DC1-DC2 or DC1-DC3) and of the data center resources within DC2 or DC3.

The Cloud Orchestrator needs to make a decision for optimal connection based on TE Network constraints and data centers

resources. It may not be able to make this decision because it has only an abstract view of the TE network (as in use case in 2.1).

The cloud orchestrator can request to the TE domain controller to compute the cost of the possible TE paths (e.g., DC1-DC2 and DC1-DC3) and to the DC controller to provide the information it needs about the required data center resources within DC2 and DC3 and then it can take the decision about the optimal solution based on this information and its policy.

3. Motivations

This section provides the motivation for the YANG model defined in this document.

Section 3.1 describes the motivation for a YANG model to request path computation.

Section 3.2 describes the motivation for a YANG model which complements the TE Topology YANG model defined in [TE-TOPO].

Section 3.3 describes the motivation for a stateless YANG RPC which complements the TE Tunnel YANG model defined in [TE-TUNNEL].

3.1. Motivation for a YANG Model

3.1.1. Benefits of common data models

The YANG data model for requesting path computation is closely aligned with the YANG data models that provide (abstract) TE topology information, i.e., [TE-TOPO] as well as that are used to configure and manage TE Tunnels, i.e., [TE-TUNNEL].

There are many benefits in aligning the data model used for path computation requests with the YANG data models used for TE topology information and for TE Tunnels configuration and management:

- o There is no need for an error-prone mapping or correlation of information.
- o It is possible to use the same endpoint identifiers in path computation requests and in the topology modeling.
- o The attributes used for path computation constraints are the same as those used when setting up a TE Tunnel.

3.1.2. Benefits of a single interface

The system integration effort is typically lower if a single, consistent interface is used by controllers, i.e., one data modeling language (i.e., YANG) and a common protocol (e.g., NETCONF or RESTCONF).

Practical benefits of using a single, consistent interface include:

1. Simple authentication and authorization: The interface between different components has to be secured. If different protocols have different security mechanisms, ensuring a common access control model may result in overhead. For instance, there may be a need to deal with different security mechanisms, e.g., different credentials or keys. This can result in increased integration effort.
2. Consistency: Keeping data consistent over multiple different interfaces or protocols is not trivial. For instance, the sequence of actions can matter in certain use cases, or transaction semantics could be desired. While ensuring consistency within one protocol can already be challenging, it is typically cumbersome to achieve that across different protocols.
3. Testing: System integration requires comprehensive testing, including corner cases. The more different technologies are involved, the more difficult it is to run comprehensive test cases and ensure proper integration.
4. Middle-box friendliness: Provider and consumer of path computation requests may be located in different networks, and middle-boxes such as firewalls, NATs, or load balancers may be deployed. In such environments it is simpler to deploy a single protocol. Also, it may be easier to debug connectivity problems.
5. Tooling reuse: Implementers may want to implement path computation requests with tools and libraries that already exist in controllers and/or orchestrators, e.g., leveraging the rapidly growing eco-system for YANG tooling.

3.1.3. Extensibility

Path computation is only a subset of the typical functionality of a controller. In many use cases, issuing path computation requests comes along with the need to access other functionality on the same system. In addition to obtaining TE topology, for instance also

configuration of services (setup/modification/deletion) may be required, as well as:

1. Receiving notifications for topology changes as well as integration with fault management
2. Performance management such as retrieving monitoring and telemetry data
3. Service assurance, e.g., by triggering OAM functionality
4. Other fulfilment and provisioning actions beyond tunnels and services, such as changing QoS configurations

YANG is a very extensible and flexible data modeling language that can be used for all these use cases.

3.2. Interactions with TE Topology

The use cases described in section 2 have been described assuming that the topology view exported by each underlying SDN controller to the orchestrator is aggregated using the "virtual node model", defined in [RFC7926].

TE Topology information, e.g., as provided by [TE-TOPO], could in theory be used by an underlying SDN controllers to provide TE information to its client thus allowing a PCE available within its client to perform multi-domain path computation by its own, without requesting path computations to the underlying SDN controllers.

In case the client does not implement a PCE function, as discussed in section 1, it could not perform path computation based on TE Topology information and would instead need to request path computation to the underlying controllers to get the information it needs to compute the optimal end-to-end path.

This section analyzes the need for a client to request underlying SDN controllers for path computation even in case it implements a PCE functionality, as well as how the TE Topology information and the path computation can be complementary.

In nutshell, there is a scalability trade-off between providing all the TE information needed by PCE, when implemented by the client, to take optimal path computation decisions by its own versus sending too many requests to underlying SDN Domain Controllers to compute a set of feasible optimal intra-domain TE paths.

3.2.1. TE Topology Aggregation

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export the whole TE domain as a single abstract TE node with a "detailed connectivity matrix".

The concept of a "detailed connectivity matrix" is defined in [TE-TOPO] to provide specific TE attributes (e.g., delay, SRLGs and summary TE metrics) as an extension of the "basic connectivity matrix", which is based on the "connectivity matrix" defined in [RFC7446].

The information provided by the "detailed connectivity matrix" would be equivalent to the information that should be provided by "virtual link model" as defined in [RFC7926].

For example, in the Packet/Optical integration use case, described in section 2.1, the Optical network controller can make the information shown in Figure 3 available to the Coordinator as part of the TE Topology information and the Coordinator could use this information to calculate by its own the optimal path between R1 and R2, without requesting any additional information to the Optical network Controller.

However, when designing the amount of information to provide within the "detailed connectivity matrix", there is a tradeoff to be considered between accuracy (i.e., providing "all" the information that might be needed by the PCE available to Orchestrator) and scalability.

Figure 6 below shows another example, similar to Figure 3, where there are two possible Optical paths between VP1 and VP4 with different properties (e.g., available bandwidth and cost).

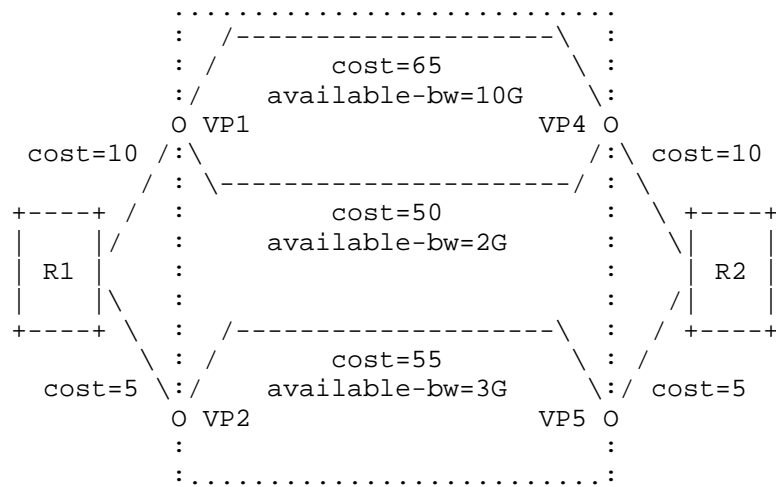


Figure 6 - Packet/Optical Path Computation Example with multiple choices

Reporting all the information, as in Figure 6, using the "detailed connectivity matrix", is quite challenging from a scalability perspective. The amount of this information is not just based on number of end points (which would scale as N-square), but also on many other parameters, including client rate, user constraints/policies for the service, e.g. max latency < N ms, max cost, etc., exclusion policies to route around busy links, min OSNR margin, max preFEC BER etc. All these constraints could be different based on connectivity requirements.

Examples of how the "detailed connectivity matrix" can be dimensioned are described in Appendix A.

It is also worth noting that the "connectivity matrix" has been originally defined in WSON, [RFC7446], to report the connectivity constraints of a physical node within the WDM network: the information it contains is pretty "static" and therefore, once taken and stored in the TE data base, it can be always being considered valid and up-to-date in path computation request.

Using the "basic connectivity matrix" with an abstract node to abstract the information regarding the connectivity constraints of an Optical domain, would make this information more "dynamic" since the connectivity constraints of an Optical domain can change over

time because some optical paths that are feasible at a given time may become unfeasible at a later time when e.g., another optical path is established. The information in the "detailed connectivity matrix" is even more dynamic since the establishment of another optical path may change some of the parameters (e.g., delay or available bandwidth) in the "detailed connectivity matrix" while not changing the feasibility of the path.

The "connectivity matrix" is sometimes confused with optical reach table that contain multiple (e.g. k-shortest) regen-free reachable paths for every A-Z node combination in the network. Optical reach tables can be calculated offline, utilizing vendor optical design and planning tools, and periodically uploaded to the Controller: these optical path reach tables are fairly static. However, to get the connectivity matrix, between any two sites, either a regen free path can be used, if one is available, or multiple regen free paths are concatenated to get from src to dest, which can be a very large combination. Additionally, when the optical path within optical domain needs to be computed, it can result in different paths based on input objective, constraints, and network conditions. In summary, even though "optical reachability table" is fairly static, which regen free paths to build the connectivity matrix between any source and destination is very dynamic, and is done using very sophisticated routing algorithms.

There is therefore the need to keep the information in the "detailed connectivity matrix" updated which means that there another tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the client's PCE) and having up-to-date information. The more the information is provided and the longer it takes to keep it up-to-date which increases the likelihood that the client's PCE computes paths using not updated information.

It seems therefore quite challenging to have a "detailed connectivity matrix" that provides accurate, scalable and updated information to allow the client's PCE to take optimal decisions by its own.

Instead, if the information in the "detailed connectivity matrix" is not complete/accurate, we can have the following drawbacks considering for example the case in Figure 6:

- o If only the VP1-VP4 path with available bandwidth of 2 Gb/s and cost 50 is reported, the client's PCE will fail to compute a 5 Gb/s path between routers R1 and R2, although this would be feasible;
- o If only the VP1-VP4 path with available bandwidth of 10 Gb/s and cost 60 is reported, the client's PCE will compute, as optimal, the 1 Gb/s path between R1 and R2 going through the VP2-VP5 path within the Optical domain while the optimal path would actually be the one going through the VP1-VP4 sub-path (with cost 50) within the Optical domain.

Using the approach proposed in this document, the client, when it needs to setup an end-to-end path, it can request the Optical domain controller to compute a set of optimal paths (e.g., for VP1-VP4 and VP2-VP5) and take decisions based on the information received:

- o When setting up a 5 Gb/s path between routers R1 and R2, the Optical domain controller may report only the VP1-VP4 path as the only feasible path: the Orchestrator can successfully setup the end-to-end path passing through this Optical path;
- o When setting up a 1 Gb/s path between routers R1 and R2, the Optical domain controller (knowing that the path requires only 1 Gb/s) can report both the VP1-VP4 path, with cost 50, and the VP2-VP5 path, with cost 65. The Orchestrator can then compute the optimal path which is passing through the VP1-VP4 sub-path (with cost 50) within the Optical domain.

3.2.2. TE Topology Abstraction

Using the TE Topology model, as defined in [TE-TOP0], the underlying SDN controller can export an abstract TE Topology, composed by a set of TE nodes and TE links, representing the abstract view of the topology controlled by each domain controller.

Considering the example in Figure 4, the TE domain controller 1 can export a TE Topology encompassing the TE nodes A, B, C and D and the TE Link interconnecting them. In a similar way, TE domain controller 2 can export a TE Topology encompassing the TE nodes E, F, G and H and the TE Link interconnecting them.

In this example, for simplicity reasons, each abstract TE node maps with each physical node, but this is not necessary.

In order to setup a multi-domain TE path (e.g., between nodes A and H), the multi-domain controller can compute by its own an optimal end-to-end path based on the abstract TE topology information provided by the domain controllers. For example:

- o Multi-domain controller's PCE, based on its own information, can compute the optimal multi-domain path being A-B-C-E-G-H, and then request the TE domain controllers to setup the A-B-C and E-G-H intra-domain paths
- o But, during path setup, the domain controller may find out that A-B-C intra-domain path is not feasible (as discussed in section 2.2, in optical networks it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller), while only the path A-B-D is feasible
- o So what the multi-domain controller computed is not good and need to re-start the path computation from scratch

As discussed in section 3.2.1, providing more extensive abstract information from the TE domain controllers to the multi-domain controller may lead to scalability problems.

In a sense this is similar to the problem of routing and wavelength assignment within an Optical domain. It is possible to do first routing (step 1) and then wavelength assignment (step 2), but the chances of ending up with a good path is low. Alternatively, it is possible to do combined routing and wavelength assignment, which is known to be a more optimal and effective way for Optical path setup. Similarly, it is possible to first compute an abstract end-to-end path within the multi-domain Orchestrator (step 1) and then compute an intra-domain path within each Optical domain (step 2), but there are more chances not to find a path or to get a suboptimal path that performing per-domain path computation and then stitch them.

3.2.3. Complementary use of TE topology and path computation

As discussed in section 2.2, there are some scalability issues with path computation requests in a multi-domain TE network with many TE domains, in terms of the number of requests to send to the TE domain controllers. It would therefore be worthwhile using the TE topology information provided by the domain controllers to limit the number of requests.

An example can be described considering the multi-domain abstract topology shown in Figure 7. In this example, an end-to-end TE path between domains A and F needs to be setup. The transit domain should be selected between domains B, C, D and E.

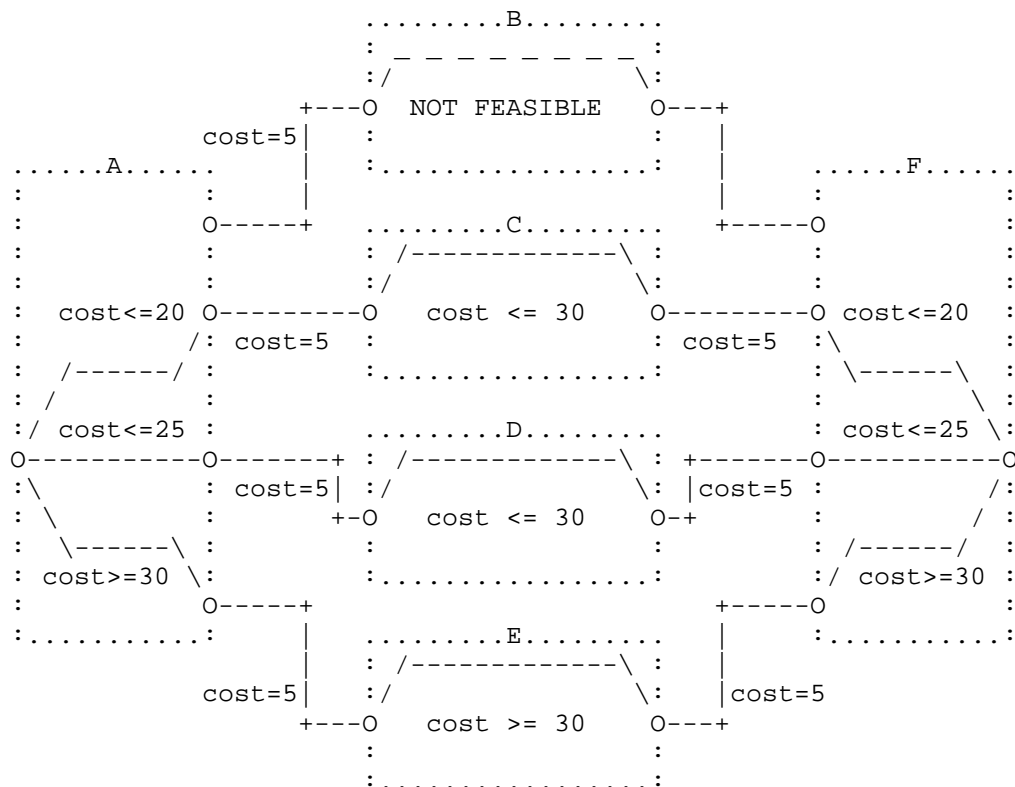


Figure 7 - Multi-domain with many domains (Topology information)

The actual cost of each intra-domain path is not known a priori from the abstract topology information. The Multi-domain controller only knows, from the TE topology provided by the underlying domain controllers, the feasibility of some intra-domain paths and some upper-bound and/or lower-bound cost information. With this information, together with the cost of inter-domain links, the Multi-domain controller can understand by its own that:

- o Domain B cannot be selected as the path connecting domains A and E is not feasible;

- o Domain E cannot be selected as a transit domain since it is known from the abstract topology information provided by domain controllers that the cost of the multi-domain path A-E-F (which is 100, in the best case) will be always be higher than the cost of the multi-domain paths A-D-F (which is 90, in the worst case) and A-E-F (which is 80, in the worst case)

Therefore, the Multi-domain controller can understand by its own that the optimal multi-domain path could be either A-D-F or A-E-F but it cannot know which one of the two possible options actually provides the optimal end-to-end path.

The Multi-domain controller can therefore request path computation only to the TE domain controllers A, D, E and F (and not to all the possible TE domain controllers).

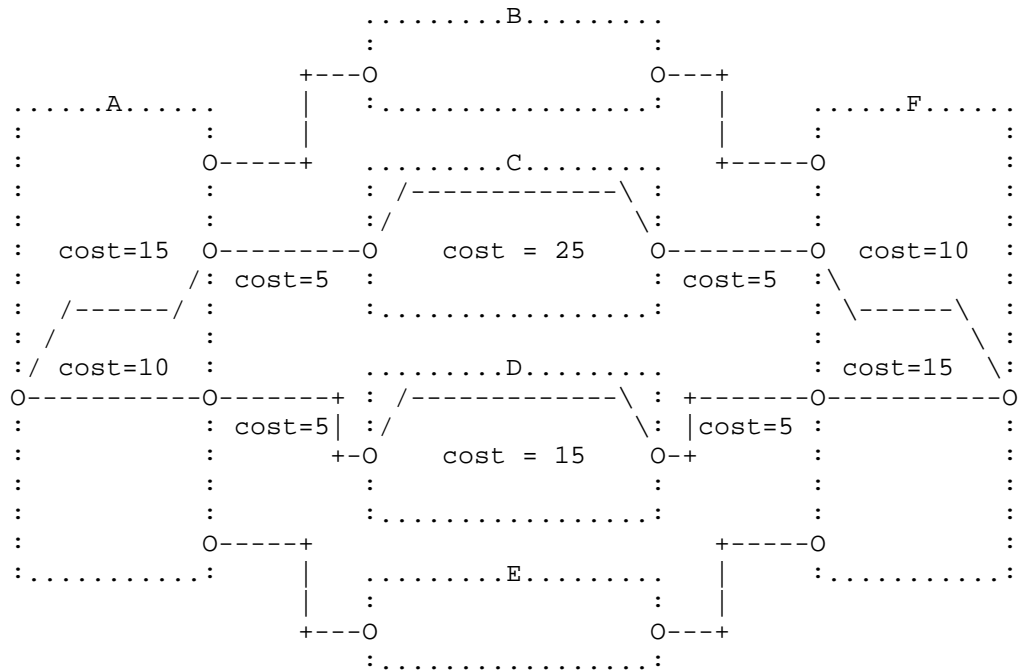


Figure 8 - Multi-domain with many domains (Path Computation information)

Based on these requests, the Multi-domain controller can know the actual cost of each intra-domain paths which belongs to potential

optimal end-to-end paths, as shown in Figure 8, and then compute the optimal end-to-end path (e.g., A-D-F, having total cost of 50, instead of A-C-F having a total cost of 70).

3.3. Stateless and Stateful Path Computation

The TE Tunnel YANG model, defined in [TE-TUNNEL], can support the need to request path computation.

It is possible to request path computation by configuring a "compute-only" TE tunnel and retrieving the computed path(s) in the LSP(s) Record-Route Object (RRO) list as described in section 3.3.1 of [TE-TUNNEL].

This is a stateful solution since the state of each created "compute-only" TE tunnel needs to be maintained and updated, when underlying network conditions change.

It is very useful to provide options for both stateless and stateful path computation mechanisms. It is suggested to use stateless mechanisms as much as possible and to rely on stateful path computation when really needed.

Stateless RPC allows requesting path computation using a simple atomic operation and it is the natural option/choice, especially with stateless PCE.

Since the operation is stateless, there is no guarantee that the returned path would still be available when path setup is requested: this is not a major issue in case the time between path computation and path setup is short (especially if compared with the time that would be needed to update the information of a very detailed connectivity matrix).

In case the stateless solution is not sufficient, a stateful solution, based on "compute-only" TE tunnel, could be used to get notifications in case the computed path has been changed.

It is worth noting that also the stateful solution, although increasing the likelihood that the computed path is available at path setup, it does not guarantee that because notifications may not be reliable or delivered on time.

The stateful path computation has also the following drawbacks:

- o Several messages required for any path computation
- o Requires persistent storage in the provider controller
- o Need for garbage collection for stranded paths
- o Process burden to detect changes on the computed paths in order to provide notifications update

4. Path Computation and Optimization for multiple paths

There are use cases, where it is advantageous to request path computation for a set of paths, through a network or through a network domain, using a single request [RFC5440].

In this case, sending a single request for multiple path computations, instead of sending multiple requests for each path computation, would reduce the protocol overhead and it would consume less resources (e.g., threads in the client and server).

In the context of a typical multi-domain TE network, there could be multiple choices for the ingress/egress points of a domain and the Multi-domain controller needs to request path computation between all the ingress/egress pairs to select the best pair. For example, in the example of section 2.2, the Multi-domain controller needs to request the TE network controller 1 to compute the A-C and the A-D paths and to the TE network controller 2 to compute the E-H and the F-H paths.

It is also possible that the Multi-domain controller receives a request to setup a group of multiple end to end connections. The multi-domain controller needs to request each TE domain controller to compute multiple paths, one (or more) for each end to end connection.

There are also scenarios where it can be needed to request path computation for a set of paths in a synchronized fashion.

One example could be computing multiple diverse paths. Computing a set of diverse paths in a not-synchronized fashion, leads to the possibility of not being able to satisfy the diversity requirement. In this case, it is preferable to compute a sub-optimal primary path for which a diversely routed secondary path exists.

There are also scenarios where it is needed to request optimizing a set of paths using objective functions that apply to the whole set of paths, see [RFC5541], e.g. to minimize the sum of the costs of all the computed paths in the set.

5. YANG Model for requesting Path Computation

This document define a YANG stateless RPC to request path computation as an "augmentation" of tunnel-rpc, defined in [TE-TUNNEL]. This model provides the RPC input attributes that are needed to request path computation and the RPC output attributes that are needed to report the computed paths.

```
augment /te:tunnels-rpc/te:input/te:tunnel-info:
  +---- path-request* [request-id]
  .....

augment /te:tunnels-rpc/te:output/te:result:
  +--ro response* [response-id]
    +--ro response-id          uint32
    +--ro (response-type)?
      +--:(no-path-case)
      |   +--ro no-path!
      +--:(path-case)
        +--ro computed-path
          +--ro path-id?          yang-types:uuid
          +--ro path-properties
          .....
  .....

```

This model extensively re-uses the grouping defined in [TE-TUNNEL] to ensure maximal syntax and semantics commonality.

5.1. Synchronization of multiple path computation requests

The YANG model permits to synchronize a set of multiple path requests (identified by specific request-id) all related to a "svec" container emulating the syntax of "SVEC" PCEP object [RFC 5440].

```
+---- synchronization* [synchronization-id]
  +---- synchronization-id  uint32
  +---- svec
    |   +---- relaxable?      boolean
    |   +---- disjointness?  te-types:te-path-disjointness

```

```

| +---- request-id-number*   uint32
+---- svec-constraints
|   +---- path-metric-bound* [metric-type]
|       +---- metric-type    identityref
|       +---- upper-bound?   uint64
+---- path-srlgs
|   +---- usage?             identityref
|   +---- values*            srlg
+---- exclude-objects
.....
+---- optimizations
    +---- (algorithm)?
        +--:(metric)
            +---- optimization-metric* [metric-type]
            |   +---- metric-type      identityref
            |   +---- weight?          uint8
        +--:(objective-function)
            +---- objective-function
                +---- objective-function-type? identityref

```

The model, in addition to the metric types, defined in [TE-TUNNEL], which can be applied to each individual path request, defines additional specific metrics types that apply to a set of synchronized requests, as referenced in [RFC5541].

```

identity svec-metric-type {
  description
    "Base identity for svec metric type";
}

identity svec-metric-cumul-te {
  base svec-metric-type;
  description
    "TE cumulative path metric";
}

identity svec-metric-cumul-igp {
  base svec-metric-type;
  description
    "IGP cumulative path metric";
}

```

```

identity svec-metric-cumul-hop {
  base svec-metric-type;
  description
    "Hop cumulative path metric";
}

identity svec-metric-aggregate-bandwidth-consumption {
  base svec-metric-type;
  description
    "Cumulative bandwith consumption of the set of synchronized
paths";
}

identity svec-metric-load-of-the-most-loaded-link {
  base svec-metric-type;
  description
    "Load of the most loaded link";
}

```

5.2. Returned metric values

This YANG model provides a way to return the values of the metrics computed by the path computation in the output of RPC, together with other important information (e.g. srlg, affinities, explicit route), emulating the syntax of the "C" flag of the "METRIC" PCEP object [RFC 5440]:

```

augment /te:tunnels-rpc/te:output/te:result:
  +--ro response* [response-id]
    +--ro response-id          uint32
    +--ro (response-type)?
      +--:(no-path-case)
      | +--ro no-path!
      +--:(path-case)
        +--ro computed-path
          +--ro path-id?          yang-types:uuid
          +--ro path-properties
            +--ro path-metric* [metric-type]
              | +--ro metric-type      identityref
              | +--ro accumulative-value?  uint64

```



```

+--ro path-affinities
|   +--ro constraint* [usage]
|       +--ro usage      identityref
|       +--ro value?     admin-groups
+--ro path-srlgs
|   +--ro usage?         identityref
|   +--ro values*        srlg
+--ro path-route-objects
.....

```

It also allows to request which metric should returned in the input of RPC:

```

augment /te:tunnels-rpc/te:input/te:tunnel-info:
+----- path-request* [request-id]
|   +----- request-id          uint32
|   .....
|   +----- requested-metrics* [metric-type]
|       +----- metric-type     identityref
|   .....

```

This feature is essential for using a stateless path computation in a multi-domain TE network as described in section 2.2. In this case, the metrics returned by a path computation requested to a given TE network controller must be used by the client to compute the best end-to-end path. If they are missing the client cannot compare different paths calculated by the TE network controllers and choose the best one for the optimal e2e path.

6. YANG model for stateless TE path computation

6.1. YANG Tree

Figure 9 below shows the tree diagram of the YANG model defined in module `ietf-te-path-computation.yang`.

```

module: ietf-te-path-computation
augment /te:tunnels-rpc/te:input/te:tunnel-info:
+----- path-request* [request-id]
|   +----- request-id          uint32
|   +----- te-topology-identifier
|       +----- provider-id?    te-types:te-global-id
|       +----- client-id?      te-types:te-global-id

```

```

| | +---- topology-id?    te-types:te-topology-id
+---- source?            inet:ip-address
+---- destination?       inet:ip-address
+---- src-tp-id?         binary
+---- dst-tp-id?         binary
+---- bidirectional?     boolean
+---- encoding?          identityref
+---- switching-type?     identityref
+---- explicit-route-objects
| | +---- route-object-exclude-always* [index]
| | | +---- index          uint32
| | | +---- (type)?
| | | | +---:(num-unnum-hop)
| | | | | +---- num-unnum-hop
| | | | | | +---- node-id?    te-types:te-node-id
| | | | | | +---- link-tp-id? te-types:te-tp-id
| | | | | | +---- hop-type?   te-hop-type
| | | | | | +---- direction?  te-link-direction
| | | | +---:(as-number)
| | | | | +---- as-number-hop
| | | | | | +---- as-number?   binary
| | | | | | +---- hop-type?   te-hop-type
| | | | +---:(label)
| | | | | +---- label-hop
| | | | | | +---- te-label
| | | | | | | +---- (technology)?
| | | | | | | | +---:(generic)
| | | | | | | | +---- generic?  rt-
types:generalized-label
| | | +---- direction?      te-label-direction
+---- route-object-include-exclude* [index]
+---- explicit-route-usage? identityref
+---- index          uint32
+---- (type)?
| | +---:(num-unnum-hop)
| | | +---- num-unnum-hop
| | | | +---- node-id?    te-types:te-node-id
| | | | +---- link-tp-id? te-types:te-tp-id
| | | | +---- hop-type?   te-hop-type

```

```

|         +---- direction?      te-link-direction
| +---:(as-number)
|   +---- as-number-hop
|     +---- as-number?    binary
|     +---- hop-type?     te-hop-type
| +---:(label)
|   +---- label-hop
|     +---- te-label
|       +---- (technology)?
|         +---:(generic)
|           +---- generic?    rt-
types:generalized-label
|         +---- direction?      te-label-direction
| +---:(srlg)
|   +---- srlg
|     +---- srlg?    uint32
+---- path-constraints
+---- te-bandwidth
|   +---- (technology)?
|     +---:(generic)
|       +---- generic?    te-bandwidth
+---- setup-priority?    uint8
+---- hold-priority?     uint8
+---- signaling-type?    identityref
+---- path-metric-bounds
|   +---- path-metric-bound* [metric-type]
|     +---- metric-type    identityref
|     +---- upper-bound?   uint64
+---- path-affinities
|   +---- constraint* [usage]
|     +---- usage          identityref
|     +---- value?         admin-groups
+---- path-srlgs
|   +---- usage?           identityref
|   +---- values*          srlg
+---- optimizations
|   +---- (algorithm)?
|     +---:(metric) {path-optimization-metric}?
|       +---- optimization-metric* [metric-type]

```

identityref					+---- metric-type	
					+---- weight?	uint8
					+---- explicit-route-exclude-objects	
					+---- route-object-exclude-object* [index]	
					+---- index	uint32
					+---- (type)?	
					+---:(num-unnum-hop)	
node-id					+---- num-unnum-hop	
					+---- node-id?	te-types:te-
tp-id					+---- link-tp-id?	te-types:te-
					+---- hop-type?	te-hop-type
direction					+---- direction?	te-link-
					+---:(as-number)	
					+---- as-number-hop	
					+---- as-number?	binary
					+---- hop-type?	te-hop-type
					+---:(label)	
					+---- label-hop	
					+---- te-label	
					+---- (technology)?	
					+---:(generic)	
types:generalized-label					+---- generic?	rt-
label-direction					+---- direction?	te-
					+---:(srlg)	
					+---- srlg	
					+---- srlg?	uint32
					+---- explicit-route-include-objects	
					+---- route-object-include-object* [index]	
					+---- index	uint32
					+---- (type)?	
					+---:(num-unnum-hop)	
					+---- num-unnum-hop	

```

node-id | | | | +---- node-id? te-types:te-
tp-id | | | | +---- link-tp-id? te-types:te-
direction | | | | +---- hop-type? te-hop-type
| | | | +---- direction? te-link-
| | | | +---:(as-number)
| | | | | +---- as-number-hop
| | | | | +---- as-number? binary
| | | | | +---- hop-type? te-hop-type
| | | | +---:(label)
| | | | | +---- label-hop
| | | | | +---- te-label
| | | | | | +---- (technology)?
| | | | | | | +---:(generic)
| | | | | | | +---- generic? rt-
types:generalized-label | | | | +---- direction? te-
label-direction | | | |
| | | | +---- tiebreakers
| | | | | +---- tiebreaker* [tiebreaker-type]
| | | | | +---- tiebreaker-type identityref
function}? +---:(objective-function) {path-optimization-objective-
| | | | +---- objective-function
| | | | | +---- objective-function-type? identityref
+---- requested-metrics* [metric-type]
| +---- metric-type identityref
+---- path-in-segment!
| +---- forward
| | +---- label-restrictions
| | | +---- label-restriction* [index]
| | | +---- restriction? enumeration
| | | +---- index uint32
| | | +---- label-start
| | | | +---- te-label
| | | | | +---- (technology)?
| | | | | | +---:(generic)

```

```

| | | | | +---- generic? rt-
types:generalized-label |
| | | | | +---- direction? te-label-direction
| | | | | +---- label-end
| | | | | | +---- te-label
| | | | | | +---- (technology)?
| | | | | | | +--:(generic)
| | | | | | | +---- generic? rt-
types:generalized-label |
| | | | | +---- direction? te-label-direction
| | | | | +---- range-bitmap? binary
| | | | | +---- reverse
| | | | | | +---- label-restrictions
| | | | | | | +---- label-restriction* [index]
| | | | | | | +---- restriction? enumeration
| | | | | | | +---- index uint32
| | | | | | | +---- label-start
| | | | | | | | +---- te-label
| | | | | | | | +---- (technology)?
| | | | | | | | | +--:(generic)
| | | | | | | | | +---- generic? rt-
types:generalized-label |
| | | | | +---- direction? te-label-direction
| | | | | +---- label-end
| | | | | | +---- te-label
| | | | | | +---- (technology)?
| | | | | | | +--:(generic)
| | | | | | | +---- generic? rt-
types:generalized-label |
| | | | | +---- direction? te-label-direction
| | | | | +---- range-bitmap? binary
| | | | | +---- path-out-segment!
| | | | | | +---- forward
| | | | | | | +---- label-restrictions
| | | | | | | | +---- label-restriction* [index]
| | | | | | | | +---- restriction? enumeration
| | | | | | | | +---- index uint32
| | | | | | | +---- label-start
| | | | | | | | +---- te-label

```

```

|
|
|
| +---- (technology)?
|   +---:(generic)
|     +---- generic?   rt-
types:generalized-label
|
|   +---- direction?   te-label-direction
|   +---- label-end
|     +---- te-label
|       +---- (technology)?
|         +---:(generic)
|           +---- generic?   rt-
types:generalized-label
|
|   +---- direction?   te-label-direction
|   +---- range-bitmap? binary
+---- reverse
  +---- label-restrictions
    +---- label-restriction* [index]
      +---- restriction?   enumeration
      +---- index          uint32
      +---- label-start
        +---- te-label
          +---- (technology)?
            +---:(generic)
              +---- generic?   rt-
types:generalized-label
|
|   +---- direction?   te-label-direction
|   +---- label-end
|     +---- te-label
|       +---- (technology)?
|         +---:(generic)
|           +---- generic?   rt-
types:generalized-label
|
|   +---- direction?   te-label-direction
|   +---- range-bitmap? binary
+---- synchronization* [synchronization-id]
  +---- synchronization-id   uint32
  +---- svec
    +---- relaxable?         boolean
    +---- disjointness?     te-types:te-path-disjointness
    +---- request-id-number* uint32

```

```

+---- svec-constraints
|   +---- path-metric-bound* [metric-type]
|   |   +---- metric-type      identityref
|   |   +---- upper-bound?    uint64
+---- path-srlgs
|   +---- usage?      identityref
|   +---- values*     srlg
+---- exclude-objects
|   +---- excludes* [index]
|   |   +---- index                                uint32
|   |   +---- (type)?
|   |   +---:(num-unnum-hop)
|   |   |   +---- num-unnum-hop
|   |   |   |   +---- node-id?      te-types:te-node-id
|   |   |   |   +---- link-tp-id?   te-types:te-tp-id
|   |   |   |   +---- hop-type?     te-hop-type
|   |   |   |   +---- direction?    te-link-direction
|   |   +---:(as-number)
|   |   |   +---- as-number-hop
|   |   |   |   +---- as-number?    binary
|   |   |   |   +---- hop-type?     te-hop-type
|   |   +---:(label)
|   |   |   +---- label-hop
|   |   |   |   +---- te-label
|   |   |   |   |   +---- (technology)?
|   |   |   |   |   |   +---:(generic)
|   |   |   |   |   |   |   +---- generic?    rt-
types:generalized-label
|   |   |   |   |   |   |   +---- direction?    te-label-direction
+---- optimizations
|   +---- (algorithm)?
|   +---:(metric)
|   |   +---- optimization-metric* [metric-type]
|   |   |   +---- metric-type      identityref
|   |   |   +---- weight?          uint8
|   +---:(objective-function)
|   |   +---- objective-function
|   |   |   +---- objective-function-type?  identityref
augment /te:tunnels-rpc/te:output/te:result:

```



```

+--ro response* [response-id]
+--ro response-id          uint32
+--ro (response-type)?
+--:(no-path-case)
|   +--ro no-path!
+--:(path-case)
+--ro computed-path
+--ro path-id?              yang-types:uuid
+--ro path-properties
+--ro path-metric* [metric-type]
|   +--ro metric-type      identityref
|   +--ro accumulative-value?  uint64
+--ro path-affinities
|   +--ro constraint* [usage]
|       +--ro usage        identityref
|       +--ro value?       admin-groups
+--ro path-srlgs
|   +--ro usage?           identityref
|   +--ro values*         srlg
+--ro path-route-objects
+--ro path-route-object* [index]
+--ro index                  uint32
+--ro (type)?
+--:(num-unnum-hop)
|   +--ro num-unnum-hop
|       +--ro node-id?      te-types:te-
node-id
|       +--ro link-tp-id?   te-types:te-
tp-id
|       +--ro hop-type?     te-hop-type
|       +--ro direction?   te-link-
direction
+--:(as-number)
|   +--ro as-number-hop
|       +--ro as-number?    binary
|       +--ro hop-type?     te-hop-type
+--:(label)
+--ro label-hop
+--ro te-label

```

	+--ro (technology)?	
	+---:(generic)	
types:generalized-label	+--ro generic?	rt-
label-direction	+--ro direction?	te-

Figure 9

- TE path computation YANG tree

6.2. YANG Module

```

<CODE BEGINS>file "ietf-te-path-computation@2018-06-19.yang"
module ietf-te-path-computation {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-path-computation";
  // replace with IANA namespace when assigned

  prefix "tepc";

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-yang-types {
    prefix "yang-types";
  }

  import ietf-te {
    prefix "te";
  }

  import ietf-te-types {
    prefix "te-types";
  }

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

  contact

```

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

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

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

";

description "YANG model for stateless TE path computation";

revision "2018-06-19" {
  description "Merging pull requests #45, #47 and #48";
  reference "YANG model for stateless TE path computation";
}

/*
 * Features
 */

feature stateless-path-computation {
  description
    "This feature indicates that the system supports
    stateless path computation.";
}

/*
 * Groupings
 */

grouping path-info {
  leaf path-id {
    type yang-types:uuid;
    config false;
    description "path-id ref.";
  }
}
```

```
    uses te-types:generic-path-properties;
    description "Path computation output information";
  }

  grouping requested-metrics-info {
    description "requested metric";
    list requested-metrics {
      key 'metric-type';
      description "list of requested metrics";
      leaf metric-type {
        type identityref {
          base te-types:path-metric-type;
        }
        description "the requested metric";
      }
    }
  }

  identity svec-metric-type {
    description
      "Base identity for svec metric type";
  }

  identity svec-metric-cumul-te {
    base svec-metric-type;
    description
      "TE cumulative path metric";
  }

  identity svec-metric-cumul-igp {
    base svec-metric-type;
    description
      "IGP cumulative path metric";
  }

  identity svec-metric-cumul-hop {
    base svec-metric-type;
    description
      "Hop cumulative path metric";
  }
```

```
    }

    identity svec-metric-aggregate-bandwidth-consumption {
      base svec-metric-type;
      description
        "Cumulative bandwith consumption of the set of synchronized
paths";
    }

    identity svec-metric-load-of-the-most-loaded-link {
      base svec-metric-type;
      description
        "Load of the most loaded link";
    }

    grouping svec-metrics-bounds_config {
      description "TE path metric bounds grouping for computing a set
of
      synchronized requests";
      leaf metric-type {
        type identityref {
          base svec-metric-type;
        }
        description "TE path metric type usable for computing a set of
      synchronized requests";
      }
      leaf upper-bound {
        type uint64;
        description "Upper bound on end-to-end svec path metric";
      }
    }

    grouping svec-metrics-optimization_config {
      description "TE path metric bounds grouping for computing a set
of
      synchronized requests";
      leaf metric-type {
        type identityref {
          base svec-metric-type;
        }
      }
    }
```

```
    }
    description "TE path metric type usable for computing a set of
        synchronized requests";
}
leaf weight {
    type uint8;
    description "Metric normalization weight";
}
}

grouping svec-exclude {
    description "List of resources to be excluded by all the paths
        in the SVEC";
    container exclude-objects {
        description "resources to be excluded";
        list excludes {
            key index;
            description
                "List of explicit route objects to always exclude
                    from synchronized path computation";
            uses te-types:explicit-route-hop;
        }
    }
}

grouping synchronization-constraints {
    description "Global constraints applicable to synchronized
        path computation";
    container svec-constraints {
        description "global svec constraints";
        list path-metric-bound {
            key metric-type;
            description "list of bound metrics";
            uses svec-metrics-bounds_config;
        }
    }
    uses te-types:generic-path-srlgs;
    uses svec-exclude;
}
```

```
grouping synchronization-optimization {
  description "Synchronized request optimization";
  container optimizations {
    description
      "The objective function container that includes
       attributes to impose when computing a synchronized set of
paths";

    choice algorithm {
      description "Optimizations algorithm.";
      case metric {
        list optimization-metric {
          key "metric-type";
          description "svec path metric type";
          uses svec-metrics-optimization_config;
        }
      }
      case objective-function {
        container objective-function {
          description
            "The objective function container that includes
             attributes to impose when computing a TE path";
          uses te-types:path-objective-function_config;
        }
      }
    }
  }
}

grouping synchronization-info {
  description "Information for sync";
  list synchronization {
    key "synchronization-id";
    description "sync list";
    leaf synchronization-id {
      type uint32;
      description "index";
    }
  }
}
```

```
    container svec {
      description
        "Synchronization VECtor";
      leaf relaxable {
        type boolean;
        default true;
        description
          "If this leaf is true, path computation process is free
to ignore svec content.
          otherwise it must take into account this svec.";
      }
      uses te-types:generic-path-disjointness;
      leaf-list request-id-number {
        type uint32;
        description "This list reports the set of M path
computation
          requests that must be synchronized.";
      }
    }
    uses synchronization-constraints;
    uses synchronization-optimization;
  }
}

grouping no-path-info {
  description "no-path-info";
  container no-path {
    presence "Response without path information, due to failure
    performing the path computation";
    description "if path computation cannot identify a path,
    rpc returns no path.";
  }
}

/*
 * These groupings should be removed when defined in te-types
 */

grouping encoding-and-switching-type {
```



```
    description
      "Common grouping to define the LSP encoding and switching
types";

    leaf encoding {
      type identityref {
        base te-types:lsp-encoding-types;
      }
      description "LSP encoding type";
      reference "RFC3945";
    }
    leaf switching-type {
      type identityref {
        base te-types:switching-capabilities;
      }
      description "LSP switching type";
      reference "RFC3945";
    }
  }

  grouping end-points {
    description
      "Common grouping to define the TE tunnel end-points";

    leaf source {
      type inet:ip-address;
      description "TE tunnel source address.";
    }
    leaf destination {
      type inet:ip-address;
      description "P2P tunnel destination address";
    }
    leaf src-tp-id {
      type binary;
      description "TE tunnel source termination point identifier.";
    }
    leaf dst-tp-id {
      type binary;
```

```
        description "TE tunnel destination termination point
identifier.";
    }
    leaf bidirectional {
        type boolean;
        default 'false';
        description "TE tunnel bidirectional";
    }
}

/**
 * AUGMENTS TO TE RPC
 */

augment "/te:tunnels-rpc/te:input/te:tunnel-info" {
    description "statelessComputeP2PPath input";
    list path-request {
        key "request-id";
        description "request-list";
        leaf request-id {
            type uint32;
            mandatory true;
            description "Each path computation request is uniquely
identified by the request-id-number.
            It must be present also in rpcs.";
        }
        uses te-types:te-topology-identifier;
        uses end-points;
        uses encoding-and-switching-type;
        uses te-types:path-route-objects;
        uses te-types:generic-path-constraints;
        uses te-types:generic-path-optimization;
        uses requested-metrics-info;
        uses te:path-access-segment-info;
    }
    uses synchronization-info;
}

augment "/te:tunnels-rpc/te:output/te:result" {
```

```
description "statelessComputeP2PPath output";
list response {
    key response-id;
    config false;
    description "response";
    leaf response-id {
        type uint32;
        description
            "The list key that has to reuse request-id-number.";
    }
    choice response-type {
        config false;
        description "response-type";
        case no-path-case {
            uses no-path-info;
        }
        case path-case {
            container computed-path {
                uses path-info;
                description "Path computation service.";
            }
        }
    }
}
}
}
}
<CODE ENDS>
```

Figure 10 - TE path computation YANG module

7. Security Considerations

This document describes use cases of requesting Path Computation using YANG models, which could be used at the ABNO Control Interface [RFC7491] and/or between controllers in ACTN [ACTN-frame]. As such, it does not introduce any new security considerations compared to the ones related to YANG specification, ABNO specification and ACTN Framework defined in [RFC7950], [RFC7491] and [ACTN-frame].

The YANG module defined in this draft is designed to be accessed via the NETCONF protocol [RFC6241] or RESTCONF protocol [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 [RFC5246].

This document also defines common data types using the YANG data modeling language. The definitions themselves have no security impact on the Internet, but the usage of these definitions in concrete YANG modules might have. The security considerations spelled out in the YANG specification [RFC7950] apply for this document as well.

The NETCONF access control model [RFC6536] 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.

Note - The security analysis of each leaf is for further study.

8. 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-path-computation

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-path-computation

namespace: urn:ietf:params:xml:ns:yang:ietf-te-path-computation

prefix: tepc

9. References

9.1. Normative References

[RFC5541] Le Roux, JL. et al., " Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, June 2009.

- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.
- [RFC7491] Farrel, A., King, D., "A PCE-Based Architecture for Application-Based Network Operations", RFC 7491, March 2015.
- [RFC7926] Farrel, A. et al., "Problem Statement and Architecture for Information Exchange Between Interconnected Traffic Engineered Networks", RFC 7926, July 2016.
- [RFC7950] Bjorklund, M., "The YANG 1.1 Data Modeling Language", RFC 7950, August 2016.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, January 2017.
- [TE-TOPO] Liu, X. et al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress.
- [TE-TUNNEL] Saad, T. et al., "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te, work in progress.
- [ACTN-Frame] Ceccarelli, D., Lee, Y. et al., "Framework for Abstraction and Control of Traffic Engineered Networks" draft-ietf-actn-framework, work in progress.
- [ACTN-Info] Lee, Y., Belotti, S., Dhody, D., Ceccarelli, D., "Information Model for Abstraction and Control of Transport Networks", draft-ietf-teas-actn-info, work in progress.

9.2. Informative References

- [RFC4655] Farrel, A. et al., "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006.
- [RFC7139] Zhang, F. et al., "GMPLS Signaling Extensions for Control of Evolving G.709 Optical Transport Networks", RFC 7139, March 2014.
- [RFC7446] Lee, Y. et al., "Routing and Wavelength Assignment Information Model for Wavelength Switched Optical Networks", RFC 7446, February 2015.

[OTN-TOPO] Zheng, H. et al., "A YANG Data Model for Optical Transport Network Topology", draft-ietf-ccamp-otn-topo-yang, work in progress.

[PCEP-Service-Aware] Dhody, D. et al., "Extensions to the Path Computation Element Communication Protocol (PCEP) to compute service aware Label Switched Path (LSP)", draft-ietf-pce-pcep-service-aware, work in progress.

[ITU-T G.709-2016] ITU-T Recommendation G.709 (06/16), "Interface for the optical transport network", June 2016.

10. Acknowledgments

The authors would like to thank Igor Bryskin and Xian Zhang for participating in discussions and providing valuable insights.

The authors would like to thank the authors of the TE Tunnel YANG model [TE-TUNNEL], in particular Igor Bryskin, Vishnu Pavan Beeram, Tarek Saad and Xufeng Liu, for their inputs to the discussions and support in having consistency between the Path Computation and TE Tunnel YANG models.

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

Appendix A. Examples of dimensioning the "detailed connectivity matrix"

In the following table, a list of the possible constraints, associated with their potential cardinality, is reported.

The maximum number of potential connections to be computed and reported is, in first approximation, the multiplication of all of them.

Constraint	Cardinality
End points	$N(N-1)/2$ if connections are bidirectional (OTN and WDM), $N(N-1)$ for unidirectional connections.
Bandwidth	<p>In WDM networks, bandwidth values are expressed in GHz.</p> <p>On fixed-grid WDM networks, the central frequencies are on a 50GHz grid and the channel width of the transmitters are typically 50GHz such that each central frequency can be used, i.e., adjacent channels can be placed next to each other in terms of central frequencies.</p> <p>On flex-grid WDM networks, the central frequencies are on a 6.25GHz grid and the channel width of the transmitters can be multiples of 12.5GHz.</p> <p>For fixed-grid WDM networks typically there is only one possible bandwidth value (i.e., 50GHz) while for flex-grid WDM networks typically there are 4 possible bandwidth values (e.g., 37.5GHz, 50GHz, 62.5GHz, 75GHz).</p> <p>In OTN (ODU) networks, bandwidth values are expressed as pairs of ODU type and, in case of ODUFlex, ODU rate in bytes/sec as described in section 5 of [RFC7139].</p> <p>For "fixed" ODUk types, 6 possible bandwidth values are possible (i.e., ODU0, ODU1, ODU2, ODU2e, ODU3, ODU4).</p> <p>For ODUFlex(GFP), up to 80 different bandwidth values can be specified, as defined in Table 7-8 of [ITU-T G.709-2016].</p> <p>For other ODUFlex types, like ODUFlex(CBR), the number of possible bandwidth values depends on the rates of the</p>

clients that could be mapped over these ODUFlex types, as shown in Table 7.2 of [ITU-T G.709-2016], which in theory could be a countinuum of values. However, since different ODUFlex bandwidths that use the same number of TSs on each link along the path are equivalent for path computation purposes, up to 120 different bandwidth ranges can be specified.

Ideas to reduce the number of ODUFlex bandwidth values in the detailed connectivity matrix, to less than 100, are for further study.

Bandwidth specification for ODUCn is currently for further study but it is expected that other bandwidth values can be specified as integer multiples of 100Gb/s.

In IP we have bandwidth values in bytes/sec. In principle, this is a countinuum of values, but in practice we can identify a set of bandwidth ranges, where any bandwidth value inside the same range produces the same path.

The number of such ranges is the cardinality, which depends on the topology, available bandwidth and status of the network. Simulations (Note: reference paper submitted for publication) show that values for medium size topologies (around 50-150 nodes) are in the range 4-7 (5 on average) for each end points couple.

Metrics IGP, TE and hop number are the basic objective metrics defined so far. There are also the 2 objective functions defined in [RFC5541]: Minimum Load Path (MLP) and Maximum Residual Bandwidth Path (MBP). Assuming that one only metric or objective function can be optimized at once, the total cardinality here is 5.

With [PCEP-Service-Aware], a number of additional metrics are defined, including Path Delay metric, Path Delay Variation metric and Path Loss metric, both for point-to-point and point-to-multipoint paths. This increases the cardinality to 8.

Bounds Each metric can be associated with a bound in order to find a path having a total value of that metric lower than the given bound. This has a potentially very high cardinality (as any value for the bound is allowed). In

practice there is a maximum value of the bound (the one with the maximum value of the associated metric) which results always in the same path, and a range approach like for bandwidth in IP should produce also in this case the cardinality. Assuming to have a cardinality similar to the one of the bandwidth (let say 5 on average) we should have 6 (IGP, TE, hop, path delay, path delay variation and path loss; we don't consider here the two objective functions of [RFC5541] as they are conceived only for optimization)*5 = 30 cardinality.

Technology

constraints For further study

Priority We have 8 values for setup priority, which is used in path computation to route a path using free resources and, where no free resources are available, resources used by LSPs having a lower holding priority.

Local prot It's possible to ask for a local protected service, where all the links used by the path are protected with fast reroute (this is only for IP networks, but line protection schemas are available on the other technologies as well). This adds an alternative path computation, so the cardinality of this constraint is 2.

Administrative

Colors Administrative colors (aka affinities) are typically assigned to links but when topology abstraction is used affinity information can also appear in the detailed connectivity matrix.

There are 32 bits available for the affinities. Links can be tagged with any combination of these bits, and path computation can be constrained to include or exclude any or all of them. The relevant cardinality is 3 (include-any, exclude-any, include-all) times 2^{32} possible values. However, the number of possible values used in real networks is quite small.

Included Resources

A path computation request can be associated to an ordered set of network resources (links, nodes) to be included along the computed path. This constraint would

have a huge cardinality as in principle any combination of network resources is possible. However, as far as the Orchestrator doesn't know details about the internal topology of the domain, it shouldn't include this type of constraint at all (see more details below).

Excluded Resources

A path computation request can be associated to a set of network resources (links, nodes, SRLGs) to be excluded from the computed path. Like for included resources, this constraint has a potentially very high cardinality, but, once again, it can't be actually used by the Orchestrator, if it's not aware of the domain topology (see more details below).

As discussed above, the Orchestrator can specify include or exclude resources depending on the abstract topology information that the domain controller exposes:

- o In case the domain controller exposes the entire domain as a single abstract TE node with his own external terminations and detailed connectivity matrix (whose size we are estimating), no other topological details are available, therefore the size of the detailed connectivity matrix only depends on the combination of the constraints that the Orchestrator can use in a path computation request to the domain controller. These constraints cannot refer to any details of the internal topology of the domain, as those details are not known to the Orchestrator and so they do not impact size of the detailed connectivity matrix exported.

- o Instead in case the domain controller exposes a topology including more than one abstract TE nodes and TE links, and their attributes (e.g. SRLGs, affinities for the links), the Orchestrator knows these details and therefore could compute a path across the domain referring to them in the constraints. The detailed connectivity matrixes, whose size need to be estimated here, are the ones relevant to the abstract TE nodes exported to the Orchestrator. These detailed connectivity matrixes and therefore their sizes, while cannot depend on the other abstract TE nodes and TE links, which are external to the given abstract node, could depend to SRLGs (and other attributes, like affinities) which could be present also in the portion of the topology represented by the abstract nodes, and therefore contribute to the size of the related detailed connectivity matrix.

We also don't consider here the possibility to ask for more than one path in diversity or for point-to-multi-point paths, which are for further study.

Considering for example an IP domain without considering SRLG and affinities, we have an estimated number of paths depending on these estimated cardinalities:

Endpoints = $N(N-1)$, Bandwidth = 5, Metrics = 6, Bounds = 20,
Priority = 8, Local prot = 2

The number of paths to be pre-computed by each IP domain is therefore $24960 * N(N-1)$ where N is the number of domain access points.

This means that with just 4 access points we have nearly 300000 paths to compute, advertise and maintain (if a change happens in the domain, due to a fault, or just the deployment of new traffic, a substantial number of paths need to be recomputed and the relevant changes advertised to the upper controller).

This seems quite challenging. In fact, if we assume a mean length of 1K for the json describing a path (a quite conservative estimate), reporting 300000 paths means transferring and then parsing more than 300 Mbytes for each domain. If we assume that 20% (to be checked) of this paths change when a new deployment of traffic occurs, we have 60 Mbytes of transfer for each domain traversed by a new end-to-end path. If a network has, let say, 20 domains (we want to estimate the load for a non-trivial domain setup) in the beginning a total

initial transfer of 6Gigs is needed, and eventually, assuming 4-5 domains are involved in mean during a path deployment we could have 240-300 Mbytes of changes advertised to the higher order controller.

Further bare-bone solutions can be investigated, removing some more options, if this is considered not acceptable; in conclusion, it seems that an approach based only on the information provided by the detailed connectivity matrix is hardly feasible, and could be applicable only to small networks with a limited meshing degree between domains and renouncing to a number of path computation features.

Contributors

Dieter Beller
Nokia
Email: dieter.beller@nokia.com

Gianmarco Bruno
Ericsson
Email: gianmarco.bruno@ericsson.com

Francesco Lazzeri
Ericsson
Email: francesco.lazzeri@ericsson.com

Young Lee
Huawei
Email: leeyoung@huawei.com

Carlo Perocchio
Ericsson
Email: carlo.perocchio@ericsson.com

Authors' Addresses

Italo Busi (Editor)
Huawei
Email: italo.busi@huawei.com

Sergio Belotti (Editor)
Nokia
Email: sergio.belotti@nokia.com

Victor Lopez
Telefonica
Email: victor.lopezalvarez@telefonica.com

Oscar Gonzalez de Dios
Telefonica
Email: oscar.gonzalezdedios@telefonica.com

Anurag Sharma
Google
Email: ansha@google.com

Yan Shi
China Unicom
Email: shiyan49@chinaunicom.cn

Ricard Vilalta
CTTC
Email: ricard.vilalta@cttc.es

Karthik Sethuraman
NEC
Email: karthik.sethuraman@necam.com

Michael Scharf
Nokia
Email: michael.scharf@nokia.com

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

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2019

X. Liu
Volta Networks
I. Bryskin
Huawei Technologies
V. Beeram
Juniper Networks
T. Saad
Cisco Systems Inc
H. Shah
Ciena
S. Litkowski
Orange Business Service
July 1, 2018

YANG Data Model for SR and SR TE Topologies
draft-ietf-teas-yang-sr-te-topo-02

Abstract

This document defines a YANG data model for Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 2, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Terminology	2
1.2. Tree Diagrams	3
2. Modeling Considerations	3
2.1. Segment Routing (SR) Topology	3
2.2. Segment Routing (SR) TE Topology	3
2.3. Relations to ietf-segment-routing	4
2.4. Topology Type Modeling	5
2.5. Topology Attributes	5
2.6. Node Attributes	5
2.7. Link Termination Point Attributes	6
2.8. Link in the Topology Model	7
3. Model Structure	7
4. YANG Module	9
5. IANA Considerations	15
6. Security Considerations	15
7. References	16
7.1. Normative References	16
7.2. Informative References	16
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations	18
A.1. SR Topology State Module	18
Authors' Addresses	21

1. Introduction

This document defines a YANG [RFC7950] data model for describing the presentations of Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology. The version of the model limits the transport type to an MPLS dataplane.

1.1. Terminology

The keywords "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].

The following terms are defined in [RFC7950] and are not redefined here:

- o augment
- o data model
- o data node

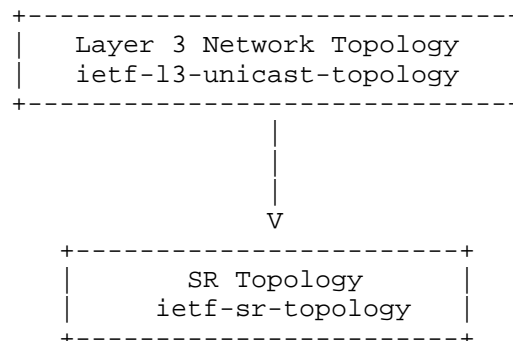
1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

2. Modeling Considerations

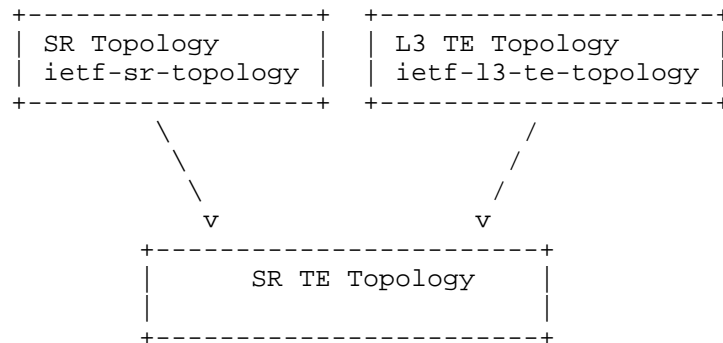
2.1. Segment Routing (SR) Topology

The Layer 3 network topology model is discussed in [RFC8346]. The Segment Routing (SR) topology model proposed in this document augments and uses the `ietf-l3-unicast-igp-topology` module defined in [RFC8346]. SR related attributes are covered in the `ietf-sr-topology` model.



2.2. Segment Routing (SR) TE Topology

When traffic engineering is enabled on an SR topology, there will be associations between objects in SR topologies and objects in TE topologies. An SR TE topology is both an SR topology and a layer 3 TE topology. Multiple inheritance is used to achieve such relations.



Each type of topologies is indicated by "network-types" defined in [RFC8345]. For the three types of topologies above, the data representations are:

L3 Topology:

```
/nd:networks/nd:network/nd:network-types/l3-unicast-topology
```

L3 TE Topology:

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

SR Topology:

```
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/sr
```

SR TE Topology: (multiple inheritance)

```
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/sr
```

2.3. Relations to ietf-segment-routing

[I-D.ietf-spring-sr-yang] defines ietf-segment-routing that is a model intended to be used on network elements to configure or operate segment routing; ietf-sr-topology defined in this document is intended to be used on a controller for the network-wide operations such as path computation.

SR topology model shares many modeling constructs defined in ietf-segment-routing. The module ietf-sr-topology uses the types and groupings defined in ietf-segment-routing.

2.4. Topology Type Modeling

A new topology type is defined in this document, to indicate a topology that is a Segment Routing (SR) topology.

```
augment /nw:networks/nw:network/nw:network-types
  /l3t:l3-unicast-topology:
    +--rw sr!
```

2.5. Topology Attributes

The Segment Routing attributes with topology-wide impacts are modeled by augmenting the container "l3-topology-attributes" in the L3 topology model. SRGB (Segment Routing Global Block) is covered in this augmentation. A SR domain is mapped to a topology in this model.

```
augment /nw:networks/nw:network/l3t:l3-topology-attributes:
  +--rw sr
    +--rw srgb* [lower-bound upper-bound]
      +--rw lower-bound    uint32
      +--rw upper-bound    uint32
```

2.6. Node Attributes

The Segment Routing attributes within the node scope are modeled by augmenting the sub tree /nw:networks/nw:network/nw:node/ in the L3 topology model.

The SR attributes that have node-scope impact are modeled by augmenting the container "l3-node-attributes" in the L3 topology model, including the SR capabilities, SRGB (Segment Routing Global Block), and SRLB (Segment Routing Local Block) specified on this mode. This model also provides the information about how these SR attributes are learned:

```

augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
  +--rw sr
    +--rw srgb* [lower-bound upper-bound]
      | +--rw lower-bound    uint32
      | +--rw upper-bound    uint32
    +--rw srlb* [lower-bound upper-bound]
      | +--rw lower-bound    uint32
      | +--rw upper-bound    uint32
    +--rw node-capabilities
      | +--rw transport-planes* [transport-plane]
      | | +--rw transport-plane  identityref
      | +--rw readable-label-stack-depth?  uint8
    +--ro information-source?      enumeration
    +--ro information-source-state
      +--ro credibility-preference?  uint16

```

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

```

augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes
  /l3t:prefix:
    +--rw sr!
      +--rw value-type?      enumeration
      +--rw start-sid        uint32
      +--rw range?          uint32
      +--rw algorithm?       identityref
      +--rw last-hop-behavior? enumeration
      | {sid-last-hop-behavior}?
      +--rw is-local?        boolean

```

2.7. Link Termination Point Attributes

A link termination point in the topology model is mapped to an interface from the Segment Routing perspective. The Adjacency Segment attributes on an interface are modeled within a link termination point. The modeling structure is as follows:

```
augment /nw:networks/nw:network/nw:node/nt:termination-point
  /l3t:l3-termination-point-attributes:
  +--rw sr!
    +--rw value-type?          enumeration
    +--rw sid                  uint32
    +--rw advertise-protection? enumeration
    +--rw is-local?           boolean
    +--ro is-backup?          boolean
    +--ro is-part-of-set?      boolean
    +--ro is-on-lan?          boolean
    +--ro information-source?  enumeration
    +--ro information-source-state
      +--ro credibility-preference? uint16
```

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

Since YANG models are usually implemented with persistent configuration datastores, this model supports only persistent Adjacency Segments.

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

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

2.8. Link in the Topology Model

A link in the topology model connects the termination point on the source node to the termination point on the destination node. When such a link is instantiated, the bindings between the nodes and the corresponding Adj-SIDs are formed, and the resulting FIB entries are installed.

3. Model Structure

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

```

module: ietf-sr-topology
  augment /nw:networks/nw:network/nw:network-types
    /l3t:l3-unicast-topology:
      +--rw sr!
  augment /nw:networks/nw:network/l3t:l3-topology-attributes:
    +--rw sr
      +--rw srgb* [lower-bound upper-bound]
        +--rw lower-bound    uint32
        +--rw upper-bound    uint32
  augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes:
    +--rw sr
      +--rw srgb* [lower-bound upper-bound]
        | +--rw lower-bound    uint32
        | +--rw upper-bound    uint32
      +--rw srlb* [lower-bound upper-bound]
        | +--rw lower-bound    uint32
        | +--rw upper-bound    uint32
      +--rw node-capabilities
        | +--rw transport-planes* [transport-plane]
        | | +--rw transport-plane  identityref
        | +--rw readable-label-stack-depth?  uint8
      +--ro information-source?      enumeration
      +--ro information-source-state
        +--ro credibility-preference?  uint16
  augment /nw:networks/nw:network/nw:node/l3t:l3-node-attributes
    /l3t:prefix:
      +--rw sr!
        +--rw value-type?      enumeration
        +--rw start-sid        uint32
        +--rw range?           uint32
        +--rw algorithm?       identityref
        +--rw last-hop-behavior? enumeration
        | {sid-last-hop-behavior}?
        +--rw is-local?        boolean
  augment /nw:networks/nw:network/nw:node/nt:termination-point
    /l3t:l3-termination-point-attributes:
      +--rw sr!
        +--rw value-type?      enumeration
        +--rw sid              uint32
        +--rw advertise-protection?  enumeration
        +--rw is-local?        boolean
        +--ro is-backup?       boolean
        +--ro is-part-of-set?   boolean
        +--ro is-on-lan?       boolean
        +--ro information-source?  enumeration
        +--ro information-source-state
          +--ro credibility-preference?  uint16

```

4. YANG Module

```
<CODE BEGINS> file "ietf-sr-topology@2018-06-22.yang"
module ietf-sr-topology {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-topology";
  prefix "srt";

  import ietf-network {
    prefix "nw";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-network-topology {
    prefix "nt";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-l3-unicast-topology {
    prefix "l3t";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }
  import ietf-segment-routing-common {
    prefix "sr-cmn";
    reference
      "I-D.ietf-spring-sr-yang: YANG Data Model for Segment Routing";
  }

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

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

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

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

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

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

     Editor:    Himanshu Shah
```


<mailto:hshah@ciena.com>

Editor: Stephane Litkowski
<mailto:stephane.litkowski@orange.com>;

description

"YANG data model for representing and manipulating Segment Routing Topologies.

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

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

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

```
revision 2018-06-22 {  
  description "Initial revision";  
  reference  
    "RFC XXXX: YANG Data Model for SR and SR TE Topologies";  
}
```

```
grouping sr-topology-type {  
  description  
    "Identifies the SR topology type.";  
  container sr {  
    presence "Indicates SR Topology";  
    description  
      "Its presence identifies the SR topology type.";  
  }  
}
```

```
augment "/nw:networks/nw:network/nw:network-types/"  
+ "l3t:l3-unicast-topology" {  
  description  
    "Defines the SR topology type.";  
  uses sr-topology-type;  
}
```

```
augment "/nw:networks/nw:network/l3t:l3-topology-attributes" {  
  when "../nw:network-types/l3t:l3-unicast-topology/srt:sr" {  
    description "Augment only for SR topology.";  
  }  
}
```

```
    }
    description "Augment topology configuration";
    uses sr-topology-attributes;
  }

  augment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes" {
    when "../..//nw:network-types/l3t:l3-unicast-topology/srt:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node configuration.";
    uses sr-node-attributes;
  }

  augment "/nw:networks/nw:network/nw:node/l3t:l3-node-attributes"
    + "/l3t:prefix" {
    when "../..//nw:network-types/l3t:l3-unicast-topology/srt:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node prefix.";
    uses sr-node-prefix-attributes;
  }

  augment "/nw:networks/nw:network/nw:node/nt:termination-point/"
    + "l3t:l3-termination-point-attributes" {
    when "../..//nw:network-types/l3t:l3-unicast-topology/"
      + "srt:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment termination point configuration";
    uses sr-tp-attributes;
  }

  grouping sr-topology-attributes {
    description "SR topology scope attributes.";
    container sr {
      description
        "Containing SR attributes.";
      uses sr-cmn:srgb-cfg;
    } // sr
  } // sr-topology-attributes

  grouping information-source-attributes {
    description
      "The attributes identifying source that has provided the
      related information, and the source credibility.";
    leaf information-source {
      type enumeration {
        enum "unknown" {
```

```
        description "The source is unknown.";
    }
    enum "locally-configured" {
        description "Configured entity.";
    }
    enum "ospfv2" {
        description "OSPFv2.";
    }
    enum "ospfv3" {
        description "OSPFv3.";
    }
    enum "isis" {
        description "ISIS.";
    }
    enum "system-processed" {
        description "System processed entity.";
    }
    enum "other" {
        description "Other source.";
    }
}
config false;
description
    "Indicates the source of the information.";
}
container information-source-state {
    config false;
    description
        "The container contains state attributes related to
        the information source.";
    leaf credibility-preference {
        type uint16;
        description
            "The preference value to calculate the traffic
            engineering database credibility value used for
            tie-break selection between different
            information-source values.
            Higher value is more preferable.";
    }
}
} // information-source-attributes

grouping sr-node-attributes {
    description "SR node scope attributes.";
    container sr {
        description
            "Containing SR attributes.";
        uses sr-cmn:srgb-cfg;
    }
}
```

```
    uses sr-cmn:srlb-cfg;
    uses sr-cmn:node-capabilities;
    // Operational state data
    uses information-source-attributes;
  } // sr
} // sr-node-attributes

grouping sr-node-prefix-attributes {
  description "Containing SR attributes for a prefix.";
  container sr {
    presence "Presence indicates SR is enabled.";
    description
      "Containing SR attributes for a prefix.";
    uses sr-cmn:prefix-sid-attributes;
    uses sr-cmn:last-hop-behavior;
    leaf is-local {
      type boolean;
      description
        "'true' if the SID is local.";
    }
  } // sr
} // sr-node-prefix-attributes

grouping sr-tp-attributes {
  description "SR termination point scope attributes";
  container sr {
    presence "Presence indicates SR is enabled.";
    description
      "Containing SR attributes.";
    uses sr-cmn:sid-value-type;
    leaf sid {
      type uint32;
      mandatory true;
      description
        "Adjacency SID, which can be either IGP-Adjacency SID
        or BGP PeerAdj SID, depending on the context.";
    }
    leaf advertise-protection {
      type enumeration {
        enum "single" {
          description
            "A single Adj-SID is associated
            with the adjacency and reflects
            the protection configuration.";
        }
        enum "dual" {
          description
            "Two Adj-SIDs will be associated
```

```
        with the adjacency if interface
        is protected. In this case
        one will be enforced with
        backup flag set, the other
        will be enforced to backup flag unset.
        In case, protection is not configured,
        a single Adj-SID will be advertised
        with backup flag unset.";
    }
}
description
    "If set, the Adj-SID refers to an
    adjacency being protected.";
}
leaf is-local {
    type boolean;
    description
        "'true' if the SID is local.";
}
leaf is-backup {
    type boolean;
    config false;
    description
        "'true' if the SID is a backup.";
}
leaf is-part-of-set {
    type boolean;
    config false;
    description
        "'true' if the SID is part of a set.";
}
leaf is-on-lan {
    type boolean;
    config false;
    description
        "'true' if on a lan.";
}
}
uses information-source-attributes;
} // sr
} // sr-tp-attributes
}
<CODE ENDS>
```

5. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

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

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

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

```
-----  
name:          ietf-sr-topology  
namespace:     urn:ietf:params:xml:ns:yang:ietf-sr-topology  
prefix:        srt  
reference:     RFC XXXX  
-----
```

```
-----  
name:          ietf-sr-topology-state  
namespace:     urn:ietf:params:xml:ns:yang:ietf-sr-topology-state  
prefix:        srt-s  
reference:     RFC XXXX  
-----
```

6. Security Considerations

The configuration, state, action and notification data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241]. The data-model by itself does not create any security implications. The security considerations for the NETCONF protocol are applicable. The NETCONF protocol used for sending the data supports authentication and encryption.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [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>>.
- [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>>.
- [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>>.

7.2. Informative References

- [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>>.
- [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>>.
- [RFC8346] Clemm, A., Medved, J., Varga, R., Liu, X., Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model for Layer 3 Topologies", RFC 8346, DOI 10.17487/RFC8346, March 2018, <<https://www.rfc-editor.org/info/rfc8346>>.
- [I-D.ietf-teas-yang-te-topo] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-18 (work in progress), June 2018.

[I-D.ietf-spring-sr-yang]

Litkowski, S., Qu, Y., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", draft-ietf-spring-sr-yang-08 (work in progress), December 2017.

Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module `ietf-sr-topology` defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module, `ietf-sr-topology-state`, is defined as state model, which mirrors the module `ietf-sr-topology` defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion module, `ietf-sr-topology-state`, is redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion module mirrors that of the cooresponding NMDA model, the YANG tree of the companion module is not depicted separately.

A.1. SR Topology State Module

```
<CODE BEGINS> file "ietf-sr-topology-state@2018-06-22.yang"
module ietf-sr-topology-state {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-topology-state";
  prefix "srt-s";

  import ietf-sr-topology {
    prefix "srt";
  }
  import ietf-network-state {
    prefix "nw-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-network-topology-state {
    prefix "nt-s";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
    reference "RFC 8346: A YANG Data Model for Layer 3 Topologies";
  }
  import ietf-segment-routing-common {
    prefix "sr-cmn";
    reference
      "I-D.ietf-spring-sr-yang: YANG Data Model for Segment Routing";
  }
}
```

```
}

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

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

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

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

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

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

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

  Editor:     Stephane Litkowski
               <mailto:stephane.litkowski@orange.com>";

description
  "YANG data model for representing operational state information
  of Segment Routing Topologies, when NMDA is not supported.

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

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

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

revision 2018-06-22 {
  description "Initial revision";
  reference
```

```
    "RFC XXXX: YANG Data Model for SR and SR TE Topologies";
  }

  augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
    + "l3t-s:l3-unicast-topology" {
    description
      "Defines the SR topology type.";
    uses srt:sr-topology-type;
  }

  augment "/nw-s:networks/nw-s:network/"
    + "l3t-s:l3-topology-attributes" {
    when "../nw-s:network-types/l3t-s:l3-unicast-topology/srt-s:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment topology configuration";
    uses srt:sr-topology-attributes;
  }

  augment "/nw-s:networks/nw-s:network/nw-s:node/"
    + "l3t-s:l3-node-attributes" {
    when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "srt-s:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node configuration.";
    uses srt:sr-node-attributes;
  }

  augment "/nw-s:networks/nw-s:network/nw-s:node/"
    + "l3t-s:l3-node-attributes/l3t-s:prefix" {
    when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "srt-s:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node prefix.";
    uses srt:sr-node-prefix-attributes;
  }

  augment "/nw-s:networks/nw-s:network/nw-s:node/"
    + "nt-s:termination-point/"
    + "l3t-s:l3-termination-point-attributes" {
    when "../nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "srt-s:sr" {
      description "Augment only for SR topology.";
    }
    description "Augment termination point configuration";
    uses srt:sr-tp-attributes;
  }
```

```
    }  
    grouping sr-topology-attributes {  
      description "SR topology scope attributes.";   
      container sr {  
        description  
          "Containing SR attributes.";   
        uses sr-cmn:srgb-cfg;  
      } // sr  
    } // sr-topology-attributes  
  }  
<CODE ENDS>
```

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
Cisco Systems Inc

EMail: tsaad@cisco.com

Himanshu Shah
Ciena

EMail: hshah@ciena.com

Stephane Litkowski
Orange Business Service

EMail: stephane.litkowski@orange.com

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2019

T. Saad, Ed.
R. Gandhi
Cisco Systems Inc
X. Liu
Jabil
V. Beeram
Juniper Networks
H. Shah
Ciena
I. Bryskin
Huawei Technologies
July 01, 2018

A YANG Data Model for Traffic Engineering Tunnels and Interfaces
draft-ietf-teas-yang-te-16

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. The model also includes module(s) that contain reusable TE data types and data groupings.

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 January 2, 2019.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Terminology	3
1.2. Tree Diagram	3
1.3. Prefixes in Data Node Names	4
1.4. TE Technology Models	5
1.5. State Data Organization	6
2. Model Overview	6
2.1. Module(s) Relationship	6
2.2. Design Considerations	8
2.3. Optional Features	9
2.4. Configuration Inheritance	9
3. TE Generic Model Organization	9
3.1. Global Configuration and State Data	10
3.2. Interfaces Configuration and State Data	11
3.3. Tunnels Configuration and State Data	12
3.3.1. Tunnel Compute-Only Mode	13
3.3.2. Tunnel Hierarchical Link Endpoint	13
3.4. TE LSPs State Data	13
3.5. Global RPC Data	14
3.6. Interface RPC Data	14
3.7. Tunnel RPC Data	14
3.8. Global Notifications Data	14
3.9. Interfaces Notifications Data	14
3.10. Tunnel Notification Data	15
4. TE Generic and Helper YANG Modules	55
5. IANA Considerations	169
6. Security Considerations	170
7. Acknowledgement	170
8. Contributors	171
9. Normative References	171
Authors' Addresses	172

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. RESTCONF [RFC8040]) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interfaces, such as CLI and programmatic APIs.

This document describes the YANG data models for TE Tunnels, Label Switched Paths (LSPs) and TE interfaces that cover data applicable to generic or device-independent, device-specific, Multiprotocol Label Switching (MPLS) technology specific, and Segment Routing (SR) TE technology. It also describes helper modules that define TE grouping(s) and data types that can be imported by other modules.

The document defines the high-level relationship between the modules defined in this document, as well as other external protocol modules. It is expected other data plane technology model(s) will augment the TE generic model. Also, the TE generic model does not include any data specific to a signaling protocol. It is expected YANG models for TE signaling protocols, such as RSVP-TE ([RFC3209], [RFC3473]), or Segment-Routing TE (SR-TE) will augment the TE generic module.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

1.2. Tree Diagram

A simplified graphical representation of the data model is presented in each section of the model. The following notations are used for the YANG model data tree representation.

<status> <flags> <name> <opts> <type>

<status> is one of:

- + for current
- x for deprecated
- o for obsolete

<flags> is one of:

- rw for read-write configuration data
- ro for read-only non-configuration data
- x for execution rpcs
- n for notifications

<name> is the name of the node

If the node is augmented into the tree from another module, its name is printed as <prefix>:<name>

<opts> is one of:

- ? for an optional leaf or node
- ! for a presence container
- * for a leaf-list or list

Brackets [<keys>] for a list's keys

Curly braces {<condition>} for optional feature that make node conditional

Colon : for marking case nodes

Ellipses ("...") subtree contents not shown

Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

<type> is the name of the type for leafs and leaf-lists.

1.3. Prefixes in Data Node Names

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

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
te	ietf-te	this document
te-types	ietf-te-types	this document
te-mpls-types	ietf-te-mpls-types	this document
te-dev	ietf-te-device	this document
te-mpls	ietf-te-mpls	this document
te-sr-mpls	ietf-te-sr-mpls	this document

Table 1: Prefixes and corresponding YANG modules

1.4. TE Technology Models

This document describes the generic TE YANG data model that is independent of any dataplane technology. One of the design objectives is to allow specific data plane technologies models to reuse the generic TE data model and possibly augment it with technology specific data model(s). There are multiple options being considered to achieve this:

- o The generic TE model, including the lists of TE tunnels, LSPs, and interfaces can be defined and rooted at the top of the YANG tree. Specific leaf(s) under the TE tunnel, LSP, or interface, in this case, can identify the specific technology layer that it belongs to. This approach implies a single list for each of TE tunnel(s), LSP(s), and interface(s) in the model carries elements of different technology layers.
- o An instance of the generic TE YANG model can be mounted in the YANG tree once for each TE technology layer(s). This approach provides separation of elements belonging to different technology layers into separate lists per layer in the data model.
- o The generic TE data node(s) and TE list(s) for tunnels, LSPs, and interfaces are defined as grouping(s) in a separate module. The specific technology layer imports the generic TE groupings and uses them their respective technology specific module.

This revision of the model leverages the LSP encoding type of a tunnel (and interfaces) to identify the specific technology associated with the a TE interfaces, tunnel(s) and the LSP(s). For example, for an MPLS TE LSP, the LSP encoding type is assumed to be "lsp-encoding-packet".

Finally, the TE generic model does not include any signaling protocol data. It is expected that TE signaling protocol module(s) will be defined in other document(s) that will cover the RSVP-TE ([RFC3209], [RFC3473]), and Segment-Routing TE (SR-TE) model and that augment the TE generic model.

1.5. State Data Organization

Pure state data (for example, ephemeral or protocol derived state objects) can be modeled using one of the options below:

- o Contained inside a read-write container, in a "state" sub-container, as shown in Figure 3
- o Contained inside a separate read-only container, for example a `lsp-state` container

The Network Management Datastore Architecture (NMDA) addresses the "OpState" that was discussed in the IETF. As per NMDA guidelines for new models and models that are not concerned with the operational state of configuration information, this revision of the draft adopts the NMDA proposal for configuration and state data of this model.

2. Model Overview

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

2.1. Module(s) Relationship

The TE generic 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 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 relevant to a specific instantiations of data plane technology exists in a separate YANG module(s) that augment the TE generic model. For example, the MPLS-TE module `"ietf-te-mpls.yang"` is defined in Figure 10 and augments the TE generic model as shown in Figure 1. Similarly, the module `"ietf-te-sr-mpls.yang"` models the Segment Routing (SR) TE specific data and augments the TE generic and MPLS-TE model(s).

The TE data relevant to a TE specific signaling protocol instantiation is outside the scope and is covered in other documents. For example, the RSVP-TE [RFC3209] YANG model augmentation of the TE model is covered in [I-D.ietf-teas-yang-rsvp], and other signaling protocol model(s) (e.g. for Segment-Routing TE) are expected to also augment the TE generic model.

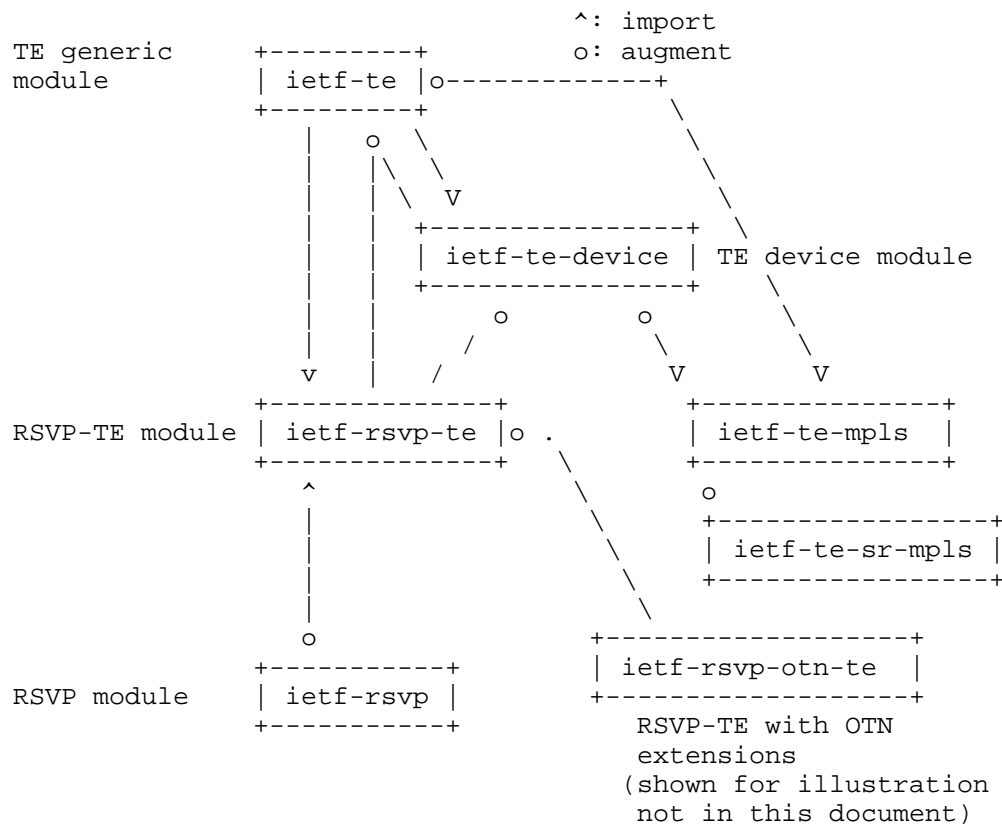


Figure 1: Relationship of TE module(s) with other signaling protocol modules

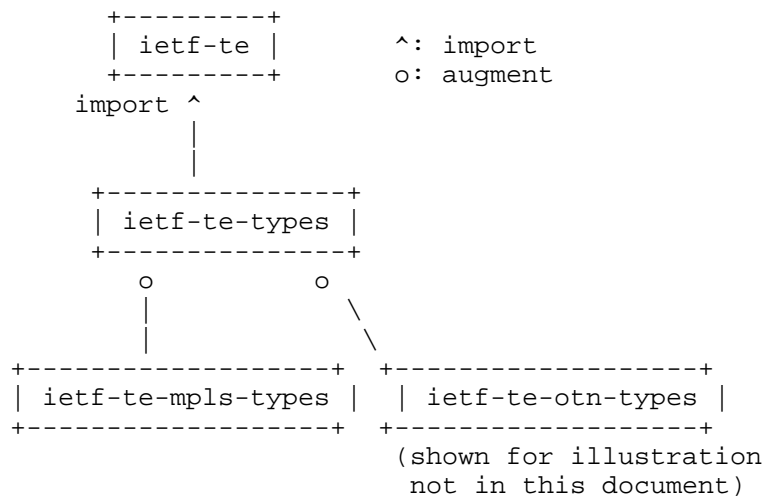


Figure 2: Relationship between generic and technology specific TE types modules

2.2. Design Considerations

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

- o reusable data elements are grouped into separate TE types module(s) that can be readily imported by other modules whenever needed
- o reusable TE data types that are data plane independent are grouped in the TE generic types module "ietf-te-types.yang"
- o reusable TE data elements that are data plane specific (e.g. packet MPLS or switching technologies as defined in [RFC3473]) are expected to be grouped in a technology- specific types module, e.g. "ietf-te-mpls-types.yang". It is expected that technology specific types will augment TE generic types as shown in Figure 2
- o The TE generic model contains device independent data and can be used to model data off a device (e.g. on a controller). The TE data that is device-specific are grouped in a separate module as shown in Figure 1.
- o In general, little information in the model is designated as "mandatory", to allow freedom to vendors to adapt the data model to their specific product implementation.

2.3. Optional Features

Optional features that are beyond the base TE model are left to the specific vendor to decide support using vendor model augmentation and/or using feature checks.

This model declares a number of TE functions as features (such as P2MP-TE, soft-preemption etc.).

2.4. Configuration Inheritance

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

3. TE Generic Model Organization

The TE generic model covers configuration, state, RPCs, and notifications data pertaining to TE global parameters, interfaces, tunnels and LSPs parameters that are device independent.

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

The model follows the guidelines in for modeling the intended, applied and derived state.

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

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

Figure 3: TE generic highlevel model view

3.1. Global Configuration and State Data

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

- o Table of named SRLG mappings
- o Table of named (extended) administrative groups mappings
- o Table of named explicit paths to be referenced by TE tunnels
- o Table of named path-constraints sets
- o Auto-bandwidth global parameters
- o TE diff-serve TE-class maps
- o System-wide capabilities for LSP reoptimization (included in the TE device model)
 - * Reoptimization timers (periodic interval, LSP installation and cleanup)
- o System-wide capabilities for TE state flooding (included in the TE device model)

- * Periodic flooding interval
- o Global capabilities that affect the originating, traversing and terminating LSPs. For example:
 - * Path selection parameters (e.g. metric to optimize, etc.)
 - * Path or segment protection parameters

The global state data is represented under the global "state" sub-container as shown in Figure 3.

Examples of such states are:

- o Global statistics (signaling, admission, preemption, flooding)
- o Global counters (number of tunnels/LSPs/interfaces)

3.2. Interfaces Configuration and State Data

This branch of the model covers configuration and state data items corresponding to TE interfaces that are present on a specific device. A new module is introduced that holds the TE device specific properties.

Examples of TE interface properties are:

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

The state corresponding to the TE interfaces applied configuration, protocol derived state, and stats and counters all fall under the interface "state" sub-container as shown in Figure 4 below:


```

module: ietf-te
  +--rw te!
    +--rw interfaces
      .
      +-- rw te-attributes
        <<intended configuration>>
      .
      +-- ro state
        <<derived state associated with the TE interface>>

```

Figure 4: TE interface state

This covers state data for TE interfaces such as:

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

3.3. Tunnels Configuration and State Data

This branch of the model covers intended, and corresponding applied configuration for tunnels. As well, it holds possible derived state pertaining to TE tunnels.

```

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

```

Figure 5: TE interface state tree

Examples of tunnel configuration data for TE tunnels:

- o Name and type (e.g. P2P, P2MP) of the TE tunnel
- o Admin-state
- o Set of primary and corresponding secondary paths
- o Routing usage (auto-route announce, forwarding adjacency)
- o Policy based routing (PBR) parameters

3.3.1. Tunnel Compute-Only Mode

By default, a configured TE tunnel is provisioned so it can carry traffic as soon as a valid path is computed and an LSP instantiated in the network. In other cases, a TE tunnel may be provisioned for computed path reporting purposes without the need to instantiate an LSP or commit resources in the network. In such a case, a tunnel configuration 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 path constraint(s) and properties on a device or controller. The device or controller is expected to compute the feasible path(s) subject to configured constraints for of "compute-only" tunnel and reflect the computed path(s) in the LSP(s) Record-Route Object (RRO) list. 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 TE tunnel is associated with.

3.4. TE LSPs State Data

TE LSPs are derived state data that is usually instantiated via signaling protocols. TE LSPs exists on routers as ingress (starting point of LSP), transit (mid-point of LSP), or egress (termination point of the LSP). TE LSPs are distinguished by the 5 tuple, and LSP type (P2P or P2MP). In the model, the nodes holding LSPs data exist in the read-only lsp-state list as show in Figure 6.

3.5. Global RPC Data

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

- o Clear global TE statistics of various features

3.6. Interface RPC Data

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

- o Clear TE statistics for all or for individual TE interfaces
- o Trigger immediate flooding for one or all TE interfaces

3.7. Tunnel RPC Data

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

- o Clear statistics for all or for individual tunnels
- o Trigger the tear and setup of existing tunnels or LSPs.

3.8. Global Notifications Data

This branch of the model covers system-wide notifications data. The node notifies the registered events to the server using the defined notification messages.

3.9. Interfaces Notifications Data

This branch of the model covers TE interfaces related notifications data. The TE interface configuration is used for specific events registration. Notifications are sent for registered events to the server. Example events for TE interfaces are:

- o Interface creation and deletion
- o Interface state transitions
- o (Soft) preemption triggers
- o Fast reroute activation

3.10. Tunnel Notification Data

This branch of the model covers TE tunnels related notifications data. The TE tunnels configuration is used for specific events registration. Notifications are sent for registered events to the server. Example events for TE tunnels are:

- o Tunnel creation and deletion events
- o Tunnel state up/down changes
- o Tunnel state reoptimization changes

Figure Figure 6 below shows the tree diagram of the YANG model defined in modules: ietf-te.yang, ietf-te-device.yang, ietf-te-mppls.yang, and ietf-te-sr.yang.

```

module: ietf-te
+--rw te!
  +--rw globals
  |   +--rw named-admin-groups
  |   |   +--rw named-admin-group* [name]
  |   {te-types:extended-admin-groups,te-types:
  |   named-extended-admin-groups}?
  |   |   +--rw name string
  |   |   +--rw bit-position? uint32
  |   +--rw named-srlgs
  |   |   +--rw named-srlg* [name] {te-types:named-srlg-groups}?
  |   |   +--rw name string
  |   |   +--rw group? te-types:srlg
  |   |   +--rw cost? uint32
  |   +--rw named-path-constraints
  |   |   +--rw named-path-constraint* [name]
  |   {te-types:named-path-constraints}?
  |   |   +--rw name string
  |   +--rw te-bandwidth
  |   |   +--rw (technology)?
  |   |   |   +--:(generic)
  |   |   |   +--rw generic? te-bandwidth
  |   +--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
  |   |   +--rw constraints* [usage]

```

```

    +---rw usage                                identityref
    +---rw (style)?
    +---:(value)
    |   +---rw value?                            te-types:admin-groups
    +---:(named)
    +---rw affinity-names* [name]
    +---rw name                                string
+---rw path-srlgs
+---rw (style)?
+---:(values)
|   +---rw usage?                                identityref
|   +---rw values*                            te-types:srlg
+---:(named)
+---rw constraints
+---rw constraint* [usage]
+---rw usage                                identityref
+---rw constraint
+---rw srlg-names* [name]
+---rw name                                string
+---rw explicit-route-objects
+---rw route-object-exclude-always* [index]
+---rw index                                uint32
+---rw (type)?
+---:(num-unnum-hop)
|   +---rw num-unnum-hop
|   +---rw node-id?                            te-types:te-node-id
|   +---rw link-tp-id?                        te-types:te-tp-id
|   +---rw hop-type?                          te-hop-type
|   +---rw direction?                        te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|   +---rw as-number?                        binary
|   +---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)?
+---:(num-unnum-hop)
|   +---rw num-unnum-hop

```

```

+---rw node-id?          te-types:te-node-id
+---rw link-tp-id?       te-types:te-tp-id
+---rw hop-type?         te-hop-type
+---rw direction?        te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|   +---rw as-number?     binary
|   +---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*  te:tunnel-ref
+---rw path-in-segment!
|   +---rw forward
|   +---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 range-bitmap?     binary
+---rw reverse
+---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 range-bitmap?    binary
+--rw path-out-segment!
|   +--rw forward
|       +--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 range-bitmap?    binary
+--rw reverse
|   +--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 range-bitmap? binary
| | | | | +---ro state
| | | | | +---ro bandwidth-generic_state? te-types:te-bandwidth
| | | | | +---ro disjointness_state?
te-types:te-path-disjointness
| | | | | +---rw te-mpls:bandwidth
| | | | | +---rw te-mpls:specification-type?
te-mpls-types:te-bandwidth-type
| | | | | +---rw te-mpls:set-bandwidth?
te-mpls-types:bandwidth-kbps
| | | | | +---rw te-mpls:class-type?
te-types:te-ds-class
| | | | | +---ro te-mpls:state
| | | | | +---ro te-mpls:signaled-bandwidth?
te-mpls-types:bandwidth-kbps
| | | | | +---rw te-sr-mpls:sid-selection-mode?
te-sid-selection-mode
| | | | | +---rw te-sr-mpls:sid-protection? identityref
+---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]
| | +---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? inet:ip-address
| | +---rw destination? inet:ip-address

```



```

+--rw src-tp-id?                               binary
+--rw dst-tp-id?                               binary
+--rw bidirectional?                           boolean
+--rw association-objects
|   +--rw association-object* [type ID source global-source]
|   |   +--rw type                             identityref
|   |   +--rw ID                               uint16
|   |   +--rw source                           inet:ip-address
|   |   +--rw global-source                     inet:ip-address
|   +--rw association-object-extended* [type ID source
global-source extended-ID]
|   |   +--rw type                             identityref
|   |   +--rw ID                               uint16
|   |   +--rw source                           inet:ip-address
|   |   +--rw global-source                     inet:ip-address
|   |   +--rw extended-ID                       binary
+--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-types:te-global-id
|   +--rw client-id?         te-types:te-global-id
|   +--rw topology-id?       te-types:te-topology-id
+--rw te-bandwidth
|   +--rw (technology)?
|   |   +--:(generic)
|   |   |   +--rw generic?   te-bandwidth
+--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-types:te-global-id
+--rw client-id?                 te-types:te-global-id
+--rw topology-id?               te-types:te-topology-id
+--ro state
+--ro operational-state?          identityref
+--ro te-dev:lsp-install-interval? uint32
+--ro te-dev:lsp-cleanup-interval? uint32
+--ro te-dev:lsp-invalidation-interval? uint32
+--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
+--rw path-scope?                 identityref
+--rw optimizations
+--rw (algorithm)?
+--:(metric) {path-optimization-metric}?
+--rw optimization-metric* [metric-type]
+--rw metric-type
identityref
uint8
+--rw weight?
+--rw explicit-route-exclude-objects
+--rw route-object-exclude-object*
[index]
+--rw index                        uint32
+--rw (type)?
+--:(num-unnum-hop)
+--rw num-unnum-hop
+--rw node-id?
te-types:te-node-id
+--rw link-tp-id?
te-types:te-tp-id
+--rw hop-type?
te-hop-type
+--rw direction?
te-link-direction
+--:(as-number)
+--rw as-number-hop

```

[illegible]

```

path-optimization-objective-function}?
    +--rw objective-function
    +--rw objective-function-type?
identityref
    +--rw preference?                               uint8
    +--rw named-path-constraint?                     ->
../../../../../../../../globals/named-path-constraints/
named-path-constraint/name
{te-types:named-path-constraints}?
    +--rw te-bandwidth
    |   +--rw (technology)?
    |   |   +--:(generic)
    |   |   +--rw generic?      te-bandwidth
    +--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
    |   +--rw constraints* [usage]
    |   |   +--rw usage                                identityref
    |   |   +--rw (style)?
    |   |   |   +--:(value)
    |   |   |   |   +--rw value?
    te-types:admin-groups
    |   +--:(named)
    |   |   +--rw affinity-names* [name]
    |   |   |   +--rw name      string
    +--rw path-srlgs
    |   +--rw (style)?
    |   |   +--:(values)
    |   |   |   +--rw usage?      identityref
    |   |   |   +--rw values*    te-types:srlg
    |   +--:(named)
    |   |   +--rw constraints
    |   |   |   +--rw constraint* [usage]
    |   |   |   |   +--rw usage      identityref
    |   |   |   |   +--rw constraint
    |   |   |   |   |   +--rw srlg-names* [name]
    |   |   |   |   |   |   +--rw name      string
    +--rw explicit-route-objects
    |   +--rw route-object-exclude-always* [index]
    |   |   +--rw index                uint32
    |   +--rw (type)?
    |   |   +--:(num-unnum-hop)
    |   |   |   +--rw num-unnum-hop

```

```

+---rw node-id?          te-types:te-node-id
+---rw link-tp-id?       te-types:te-tp-id
+---rw hop-type?         te-hop-type
+---rw direction?        te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|   +---rw as-number?     binary
|   +---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)?
|   |   +---:(num-unnum-hop)
|   |   |   +---rw num-unnum-hop
|   |   |   +---rw node-id?          te-types:te-node-id
|   |   |   +---rw link-tp-id?       te-types:te-tp-id
|   |   |   +---rw hop-type?         te-hop-type
|   |   |   +---rw direction?        te-link-direction
|   |   +---:(as-number)
|   |   |   +---rw as-number-hop
|   |   |   +---rw as-number?         binary
|   |   |   +---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*  te:tunnel-ref
+---rw path-in-segment!
|   +---rw forward
|   +---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 range-bitmap?    binary
|       +---rw reverse
|           +---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 range-bitmap?    binary
|       +---rw path-out-segment!
|           +---rw forward
|               +---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 range-bitmap?    binary
|   +---rw reverse
|       +---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 range-bitmap?    binary
|   +---ro state
|       +---ro path-properties
|           +---ro path-metric* [metric-type]
|               +---ro metric-type      -> ../state/metric-type
|               +---ro state
|                   +---ro metric-type?          identityref
|                   +---ro accumulative-value?    uint64
|               +---ro path-affinities
|                   +---ro constraints* [usage]
```

```

+---ro usage                                identityref
+---ro (style)?
+---:(value)
|   +---ro value?

te-types:admin-groups

+---:(named)
+---ro affinity-names* [name]
+---ro name            string

+---ro path-srlgs
+---ro (style)?
+---:(values)
|   +---ro usage?            identityref
|   +---ro values*          te-types:srlg
+---:(named)
+---ro constraints
+---ro constraint* [usage]
+---ro usage              identityref
+---ro constraint
+---ro srlg-names* [name]
+---ro name              string

+---ro path-route-objects
+---ro path-computed-route-object* [index]
+---ro index              -> ../state/index
+---ro state
+---ro index?              uint32
+---ro (type)?
+---:(num-unnum-hop)
|   +---ro num-unnum-hop
|   +---ro node-id?

te-types:te-node-id
|   +---ro link-tp-id?

te-types:te-tp-id
|   +---ro hop-type?        te-hop-type
|   +---ro direction?

te-link-direction

+---:(as-number)
|   +---ro as-number-hop
|   +---ro as-number?      binary
|   +---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*
te:tunnel-ref
| | | | | +---ro lspd
| | | | | +---ro lsp* [source destination tunnel-id lsp-id
extended-tunnel-id]
| | | | | +---ro source
inet:ip-address
| | | | | +---ro destination
inet:ip-address
| | | | | +---ro tunnel-id
uint16
| | | | | +---ro lsp-id
uint16
| | | | | +---ro extended-tunnel-id
inet:ip-address
| | | | | +---ro 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?
te:tunnel-ref
| | | | | +---ro lsp-record-route-subobjects
| | | | |   +---ro record-route-subobject* [index]
| | | | |     +---ro index uint32
| | | | |     +---ro (type)?
| | | | |       +---:(numbered)
| | | | |       | +---ro address?
te-types:te-tp-id
| | | | |   +---ro ip-flags? binary
| | | | |   +---:(unnumbered)
| | | | |   +---ro node-id?

```

	te-types:te-node-id		+--ro link-tp-id?
	te-types:te-tp-id		+---:(label)
			+--ro label-hop
			+--ro te-label
			+--ro (technology)?
			+---:(generic)
			+--ro generic?
	rt-types:generalized-label		+--ro direction?
	te-label-direction		+--ro label-flags? binary
			+--ro path-properties
			+--ro path-metric* [metric-type]
			+--ro metric-type ->
	../state/metric-type		+--ro state
			+--ro metric-type?
	identityref		+--ro accumulative-value? uint64
			+--ro path-affinities
			+--ro constraints* [usage]
			+--ro usage
	identityref		+--ro (style)?
			+---:(value)
			+--ro value?
	te-types:admin-groups		+---:(named)
			+--ro affinity-names* [name]
			+--ro name string
			+--ro path-srlgs
			+--ro (style)?
			+---:(values)
			+--ro usage? identityref
			+--ro values* te-types:srlg
			+---:(named)
			+--ro constraints
			+--ro constraint* [usage]
			+--ro usage
	identityref		+--ro constraint
			+--ro srlg-names* [name]
			+--ro name string
			+--ro path-route-objects
			+--ro path-computed-route-object*
[index]			

```

+---ro index      -> ../state/index
+---ro state
+---ro index?

uint32

+---ro (type)?
+---:(num-unnum-hop)
|   +---ro num-unnum-hop
|   +---ro node-id?
|
|   +---ro link-tp-id?
|
|   +---ro hop-type?
|
|   +---ro direction?
|
+---:(as-number)
|   +---ro as-number-hop
|   +---ro as-number?    binary
|   +---ro hop-type?
|
+---:(label)
+---ro label-hop
+---ro te-label
+---ro (technology)?
|   +---:(generic)
|   +---ro generic?
|
+---ro direction?

+---ro shared-resources-tunnels
+---ro lsp-shared-resources-tunnel*

te: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
+---ro te-mpls:static-lsp-name?
mpls-static:static-lsp-ref
+---rw p2p-reverse-primary-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
+---rw 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)?
+---:(num-unnum-hop)
+---rw num-unnum-hop
+---rw node-id?
te-types:te-node-id
+---rw link-tp-id?
te-types:te-tp-id
+---rw hop-type?
te-hop-type
+---rw direction?
te-link-direction
+---:(as-number)
+---rw as-number-hop
+---rw as-number? binary
+---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)?
| | +---:(num-unnum-hop)
| | | +---rw num-unnum-hop
| | | +---rw node-id?
te-types:te-node-id | | +---rw link-tp-id?
te-types:te-tp-id | | +---rw hop-type?
te-hop-type | | +---rw direction?
te-link-direction | | +---:(as-number)
| | | +---rw as-number-hop
| | | +---rw as-number? binary
| | | +---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? ->
../../../../../../globals/named-pathConstraints/
namedPathConstraintName
{teTypes:namedPathConstraints}?
```

```

+---rw te-bandwidth
+---rw (technology)?
+---:(generic)
+---rw generic?      te-bandwidth
+---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
+---rw constraints* [usage]
+---rw usage      identityref
+---rw (style)?
+---:(value)
+---rw value?

te-types:admin-groups
+---:(named)
+---rw affinity-names* [name]
+---rw name      string
+---rw path-srlgs
+---rw (style)?
+---:(values)
+---rw usage?      identityref
+---rw values*      te-types:srlg
+---:(named)
+---rw constraints
+---rw constraint* [usage]
+---rw usage      identityref
+---rw constraint
+---rw srlg-names* [name]
+---rw name      string
+---rw explicit-route-objects
+---rw route-object-exclude-always* [index]
+---rw index      uint32
+---rw (type)?
+---:(num-unnum-hop)
+---rw num-unnum-hop
+---rw node-id?

te-types:te-node-id
+---rw link-tp-id?      te-types:te-tp-id
+---rw hop-type?      te-hop-type
+---rw direction?      te-link-direction
+---:(as-number)
+---rw as-number-hop
+---rw as-number?      binary
+---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)?
+---:(num-unnum-hop)
|   +---rw num-unnum-hop
|       +---rw node-id?
te-types:te-node-id
|   +---rw link-tp-id?    te-types:te-tp-id
|   +---rw hop-type?      te-hop-type
|   +---rw direction?     te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|   +---rw as-number?     binary
|   +---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*
te:tunnel-ref
+---rw path-in-segment!
+---rw forward
|   +---rw label-restrictions
|       +---rw label-restriction* [index]
|           +---rw restriction?    enumeration
|           +---rw index            uint32
|       +---rw label-start
|           +---rw te-label
|               +---rw (technology)?

```

[illegible]

[illegible]

te-types:admin-groups	<pre> +---:(named) +---ro affinity-names* [name] +---ro name string +---ro path-srlgs +---ro (style)? +---:(values) +---ro usage? identityref +---ro values* te-types:srlg +---:(named) +---ro constraints +---ro constraint* [usage] +---ro usage identityref +---ro constraint +---ro srlg-names* [name] +---ro name string +---ro path-route-objects +---ro path-computed-route-object* [index] +---ro index -> ../state/index +---ro state +---ro index? uint32 +---ro (type)? +---:(num-unnum-hop) +---ro num-unnum-hop +---ro node-id? +---ro link-tp-id? +---ro hop-type? +---ro direction? +---:(as-number) +---ro as-number-hop +---ro as-number? binary +---ro hop-type? +---:(label) +---ro label-hop +---ro te-label +---ro (technology)? +---:(generic) +---ro generic? +---ro direction? +---ro shared-resources-tunnels +---ro lsp-shared-resources-tunnel* </pre>
te-types:te-node-id	
te-types:te-tp-id	
te-hop-type	
te-link-direction	
te-hop-type	
rt-types:generalized-label	
te-label-direction	

```

te:tunnel-ref          +---ro lsp
                        +---ro lsp* [source destination tunnel-id
lsp-id extended-tunnel-id]
                        +---ro source
inet:ip-address        +---ro destination
inet:ip-address        +---ro tunnel-id
uint16                 +---ro lsp-id
uint16                 +---ro extended-tunnel-id
inet:ip-address        +---ro 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?
te:tunnel-ref          +---ro lsp-record-route-subobjects
                        | +---ro record-route-subobject* [index]
                        | | +---ro index                uint32
                        | | +---ro (type)?
                        | | +---:(numbered)
                        | | | +---ro address?
te-types:te-tp-id      | | +---ro ip-flags?         binary
                        | | +---:(unnumbered)
                        | | +---ro node-id?
te-types:te-node-id    | | +---ro link-tp-id?

```

te-types:te-tp-id		+++:(label)
		+++ro label-hop
		+++ro te-label
		+++ro (technology)?
		+++:(generic)
		+++ro generic?
rt-types:generalized-label		+++ro direction?
te-label-direction		+++ro label-flags? binary
		+++ro path-properties
		+++ro path-metric* [metric-type]
../state/metric-type		+++ro metric-type ->
		+++ro state
		+++ro metric-type?
identityref		+++ro accumulative-value? uint64
		+++ro path-affinities
		+++ro constraints* [usage]
		+++ro usage
identityref		+++ro (style)?
		+++:(value)
		+++ro value?
te-types:admin-groups		+++:(named)
		+++ro affinity-names* [name]
		+++ro name string
		+++ro path-srlgs
		+++ro (style)?
		+++:(values)
		+++ro usage?
identityref		+++ro values*
te-types:srlg		+++:(named)
		+++ro constraints
		+++ro constraint* [usage]
		+++ro usage
identityref		+++ro constraint
		+++ro srlg-names*
[name]		+++ro name
string		+++ro path-route-objects

```

| | | | | +---ro path-computed-route-object*
[index] | | | | |
| | | | | +---ro index -> ../state/index
| | | | | +---ro state
| | | | | +---ro index?
uint32 | | | | |
| | | | | +---ro (type)?
| | | | | +---:(num-unnum-hop)
| | | | | | +---ro num-unnum-hop
| | | | | | +---ro node-id?
te-types:te-node-id | | | | |
| | | | | | +---ro link-tp-id?
te-types:te-tp-id | | | | |
| | | | | | +---ro hop-type?
te-hop-type | | | | |
| | | | | | +---ro direction?
te-link-direction | | | | |
| | | | | +---:(as-number)
| | | | | | +---ro as-number-hop
| | | | | | +---ro as-number?
binary | | | | |
| | | | | | +---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*
te:tunnel-ref | | | | |
| | | | | +---rw p2p-reverse-secondary-path
| | | | | +---rw secondary-path? ->
../.../.../.../p2p-secondary-paths/p2p-secondary-path/name
| | | | | +---rw path-setup-protocol? identityref
| | | | | +---rw candidate-p2p-secondary-paths
| | | | | +---rw candidate-p2p-secondary-path* [secondary-path]
| | | | | +---rw secondary-path ->
../.../.../.../p2p-secondary-paths/p2p-secondary-path/name
| | | | | +---rw path-setup-protocol? identityref
| | | | | +---ro state
| | | | | +---ro active? boolean
| | | | | +---rw te-mpls:static-lsp-name?
mpls-static:static-lsp-ref

```

```

| | | +---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
| | |   +---rw 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)?
| | |     +---:(num-unnum-hop)
| | |       +---rw num-unnum-hop
| | |       +---rw node-id?
te-types:te-node-id
| | |   +---rw link-tp-id?
te-types:te-tp-id
| | |   +---rw hop-type?
te-hop-type
| | |   +---rw direction?
te-link-direction
| | |   +---:(as-number)
| | |     +---rw as-number-hop
| | |     +---rw as-number? binary
| | |     +---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
| | |   +---:(srlq)

```

```

+---rw srlg
+---rw srlg? uint32
+---rw explicit-route-include-objects
+---rw route-object-include-object*

[index]
+---rw index uint32
+---rw (type)?
+---:(num-unnum-hop)
| +---rw num-unnum-hop
| +---rw node-id?
te-types:te-node-id
| +---rw link-tp-id?
te-types:te-tp-id
| +---rw hop-type?
te-hop-type
| +---rw direction?
te-link-direction
+---:(as-number)
| +---rw as-number-hop
| +---rw as-number? binary
| +---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 named-path-constraint? ->
../../../../../../../../globals/named-path-constraints/
named-path-constraint/name
{te-types:named-path-constraints}?
+---rw te-bandwidth
+---rw (technology)?
+---:(generic)
+---rw generic? te-bandwidth

```

```

+--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
|   +--rw constraints* [usage]
|       +--rw usage              identityref
|       +--rw (style)?
|           +--:(value)
|               +--rw value?
te-types:admin-groups
|   +--:(named)
|       +--rw affinity-names* [name]
|           +--rw name         string
+--rw path-srpls
|   +--rw (style)?
|       +--:(values)
|           +--rw usage?          identityref
|           +--rw values*         te-types:srlg
|       +--:(named)
|           +--rw constraints
|               +--rw constraint* [usage]
|                   +--rw usage      identityref
|                   +--rw constraint
|                       +--rw srlg-names* [name]
|                           +--rw name    string
+--rw explicit-route-objects
|   +--rw route-object-exclude-always* [index]
|       +--rw index                  uint32
|       +--rw (type)?
|           +--:(num-unnum-hop)
|               +--rw num-unnum-hop
|                   +--rw node-id?      te-types:te-node-id
|                   +--rw link-tp-id?   te-types:te-tp-id
|                   +--rw hop-type?     te-hop-type
|                   +--rw direction?    te-link-direction
|           +--:(as-number)
|               +--rw as-number-hop
|                   +--rw as-number?    binary
|                   +--rw hop-type?     te-hop-type
|           +--:(label)
|               +--rw label-hop
|                   +--rw te-label
|                       +--rw (technology)?
|                           +--:(generic)

```



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

```
|      |      |      |      |      |      |  
rt-types:generalized-label          |--rw generic?  
|      |      |      |      |      |      |  
te-label-direction                  |--rw direction?  
  
+--:(srlg)  
    |--rw srlg  
        |--rw srlg?               uint32  
+--rw shared-resources-tunnels  
    |--rw lsp-shared-resources-tunnel*   te:tunnel-ref  
+--rw path-in-segment!  
    |--rw forward  
        |--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 generic?  
|      |      |      |      |      |      |  
te-label-direction                  |--rw direction?  
  
+--rw label-end  
    |--rw te-label
```

[illegible]

```
rt-types:generalized-label
|                                     |      +---rw direction?
te-label-direction
|                                     |      +---rw range-bitmap?    binary
|                                     |      +---rw reverse
|                                     |      +---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 range-bitmap?    binary
|                                     |      +---rw disjointness?
te-types:te-path-disjointness
|      +---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 state
|      +---ro path-properties
|          +---ro path-metric* [metric-type]
|              +---ro metric-type        -> ../state/metric-type
|              +---ro state
```

```

        +---ro metric-type?          identityref
        +---ro accumulative-value?   uint64
    +---ro path-affinities
        +---ro constraints* [usage]
        +---ro usage                  identityref
        +---ro (style)?
            +---:(value)
                +---ro value?
te-types:admin-groups
        +---:(named)
            +---ro affinity-names* [name]
                +---ro name          string
    +---ro path-srlgs
        +---ro (style)?
            +---:(values)
                +---ro usage?          identityref
                +---ro values*          te-types:srlg
            +---:(named)
                +---ro constraints
                    +---ro constraint* [usage]
                        +---ro usage      identityref
                        +---ro constraint
                            +---ro srlg-names* [name]
                                +---ro name      string
    +---ro path-route-objects
        +---ro path-computed-route-object* [index]
        +---ro index      -> ../state/index
        +---ro state
            +---ro index?          uint32
            +---ro (type)?
                +---:(num-unnum-hop)
                    +---ro num-unnum-hop
                        +---ro node-id?
te-types:te-node-id
                    +---ro link-tp-id?
te-types:te-tp-id
                    +---ro hop-type?      te-hop-type
                    +---ro direction?
te-link-direction
            +---:(as-number)
                +---ro as-number-hop
                    +---ro as-number?      binary
                    +---ro hop-type?        te-hop-type
            +---:(label)
                +---ro label-hop
                    +---ro te-label
                        +---ro (technology)?
                            +---:(generic)

```

```
rt-types:generalized-label | +--ro generic?
| | | | | +--ro direction?
te-label-direction
| | | | | +--ro shared-resources-tunnels
| | | | | +--ro lsp-shared-resources-tunnel*
te:tunnel-ref
| | | | | +--ro lsps
| | | | | +--ro lsp* [source destination tunnel-id lsp-id
extended-tunnel-id]
| | | | | +--ro source
inet:ip-address
| | | | | +--ro destination
inet:ip-address
| | | | | +--ro tunnel-id
uint16
| | | | | +--ro lsp-id
uint16
| | | | | +--ro extended-tunnel-id
inet:ip-address
| | | | | +--ro 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?
te:tunnel-ref
| | | | | +--ro lsp-record-route-subobjects
| | | | | +--ro record-route-subobject* [index]
| | | | | +--ro index uint32
| | | | | +--ro (type)?
| | | | | +--:(numbered)
| | | | | +--ro address?
```

te-types:te-tp-id	 +--ro ip-flags? binary +---:(unnumbered) +--ro node-id?
te-types:te-node-id	 +--ro link-tp-id?
te-types:te-tp-id	+---:(label) +--ro label-hop +--ro te-label +--ro (technology)? +---:(generic) +--ro generic?
rt-types:generalized-label	+--ro direction?
te-label-direction	+--ro label-flags? binary
	+--ro path-properties +--ro path-metric* [metric-type] +--ro metric-type ->
../state/metric-type	+--ro state +--ro metric-type?
identityref	+--ro accumulative-value? uint64
	+--ro path-affinities +--ro constraints* [usage] +--ro usage
identityref	+--ro (style)? +---:(value) +--ro value?
te-types:admin-groups	+---:(named) +--ro affinity-names* [name] +--ro name string
	+--ro path-srlgs +--ro (style)? +---:(values) +--ro usage? identityref +--ro values* te-types:srlg
	+---:(named) +--ro constraints +--ro constraint* [usage] +--ro usage
identityref	+--ro constraint +--ro srlg-names* [name]

[illegible]

```
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
| | | | | +---ro te-mpls:static-lsp-name?
mpls-static:static-lsp-ref
| | | | | +---rw te-mpls:static-lsp-name?
mpls-static:static-lsp-ref
| | | | | +----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? te:tunnel-ref
+---rw te-dev:lsp-install-interval? uint32
+---rw te-dev:lsp-cleanup-interval? uint32
+---rw te-dev:lsp-invalidation-interval? uint32
+---rw te-mpls:tunnel-igp-shortcut
| | +---rw te-mpls:shortcut-eligible? boolean
| | +---rw te-mpls:metric-type? identityref
| | +---rw te-mpls:metric? int32
| | +---rw te-mpls:routing-afs* inet:ip-version
+---rw te-mpls:forwarding
| | +---rw te-mpls:binding-label? rt-types:mpls-label
| | +---rw te-mpls:load-share? uint32
| | +---rw te-mpls:policy-class? uint8
+---rw te-mpls:bandwidth-mpls
| | +---rw te-mpls:specification-type?
te-mpls-types:te-bandwidth-type
| | +---rw te-mpls:set-bandwidth?
te-mpls-types:bandwidth-kbps
| | +---rw te-mpls:class-type? te-types:te-ds-class
| | +---ro te-mpls:state
| | | +---ro te-mpls:signaled-bandwidth?
te-mpls-types:bandwidth-kbps
| | +---rw te-mpls:auto-bandwidth
| | +---rw te-mpls:enabled? boolean
```



```

| | | +--rw te-mpls:min-bw?
te-mpls-types:bandwidth-kbps
| | | +---rw te-mpls:max-bw?
te-mpls-types:bandwidth-kbps
| | | +---rw te-mpls:adjust-interval?      uint32
| | | +---rw te-mpls:adjust-threshold?    te-types:percentage
| | | +---rw te-mpls:overflow
| | |   | +---rw te-mpls:enabled?          boolean
| | |   | +---rw te-mpls:overflow-threshold?
te-types:percentage
| | |   | +---rw te-mpls:trigger-event-count?  uint16
| | |   +---rw te-mpls:underflow
| | |     +---rw te-mpls:enabled?            boolean
| | |     +---rw te-mpls:underflow-threshold?
te-types:percentage
| | |       +---rw te-mpls:trigger-event-count?  uint16
+---rw tunnel-p2mp* [name]
| | +---rw name                string
| | +---rw identifier?         uint16
| | +---rw description?        string
| | +---ro state
| |   +---ro operational-state?  identityref
+---ro lsp-state
| +---ro lsp* [source destination tunnel-id lsp-id
extended-tunnel-id]
| | +---ro source              inet:ip-address
| | +---ro destination         inet:ip-address
| | +---ro tunnel-id           uint16
| | +---ro lsp-id              uint16
| | +---ro extended-tunnel-id  inet:ip-address
| | +---ro 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-subobjects
| |   +---ro record-route-subobject* [index]
| |     +---ro index            uint32
| |     +---ro (type)?
| |       +---:(numbered)
| |         | +---ro address?      te-types:te-tp-id
| |         | +---ro ip-flags?     binary
| |         +---:(unnumbered)

```

```

|         |         |   +--ro node-id?          te-types:te-node-id
|         |         |   +--ro link-tp-id?       te-types:te-tp-id
|         |         |   +--:(label)
|         |         |       +--ro label-hop
|         |         |           +--ro te-label
|         |         |               |   +--ro (technology)?
|         |         |               |       +--:(generic)
|         |         |               |           +--ro generic?
rt-types:generalized-label
|         |         |   +--ro direction?          te-label-direction
|         |         |       +--ro label-flags?     binary
|         |         |   +--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?        te-types:percentage
+--rw te-dev:threshold-specification?  enumeration
+--rw te-dev:up-thresholds*           te-types:percentage
+--rw te-dev:down-thresholds*         te-types:percentage
+--rw te-dev:up-down-thresholds*      te-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]
{te-types:extended-admin-groups,te-types:
named-extended-admin-groups}?
|      +---rw te-dev:named-admin-group    ->
../../../../../../te:globals/named-admin-groups/named-admin-group/
name
+---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    ->
../../../../../../te:globals/named-srlgs/named-srlg/name
+---rw te-dev:threshold-type?          enumeration
+---rw te-dev:delta-percentage?
te-types:percentage
+---rw te-dev:threshold-specification?  enumeration
+---rw te-dev:up-thresholds*
te-types:percentage
+---rw te-dev:down-thresholds*
te-types:percentage
+---rw te-dev:up-down-thresholds*
te-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

rpcs:
+---x globals-rpc
+---x interfaces-rpc
+---x tunnels-rpc
    +---w input
        +---w tunnel-info
            +---w (type)?
            |   +---:(tunnel-p2p)
            |   |   +---w p2p-id?    te:tunnel-ref
            |   +---:(tunnel-p2mp)
            |   |   +---w p2mp-id?    te: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 6: TE generic model configuration and state tree

4. TE Generic and Helper YANG Modules

```

<CODE BEGINS> file "ietf-te-types@2018-07-01.yang"
module ietf-te-types {

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

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

  import ietf-inet-types {
    prefix inet;
  }

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-routing-types {
    prefix "rt-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@cisco.com>

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@ericsson.com>

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

description

"This module contains a collection of generally
useful TE specific YANG data type definitions.";

revision "2018-07-01" {
 description "Latest revision of TE types";
 reference "RFC3209";
}

/*

* Identities

*/

identity association-type {
 description "Base identity for tunnel association";
 reference "RFC6780, RFC4872, RFC4873";
}

identity association-type-recovery {
 base association-type;
 description
 "Association Type Recovery used to association LSPs of
 same tunnel for recovery";
 reference "RFC4872";
}

identity association-type-resource-sharing {
 base association-type;

```
    description
      "Association Type Resource Sharing used to enable resource
      sharing during make-before-break.";
    reference "RFC4873";
  }
  identity association-type-double-sided-bidir {
    base association-type;
    description
      "Association Type Double Sided bidirectional used to associate
      two LSPs of two tunnels that are independently configured on
      either endpoint";
    reference "RFC7551";
  }
  identity association-type-single-sided-bidir {
    base association-type;
    description
      "Association Type Single Sided bidirectional used to associate
      two LSPs of two tunnels, where a tunnel is configured on one
      side/endpoint, and the other tunnel is dynamically created on
      the other endpoint";
    reference "RFC7551";
  }

  identity objective-function-type {
    description "Base objective function type";
    reference "RFC4657";
  }
  identity of-minimize-cost-path {
    base objective-function-type;
    description
      "Minimize cost of path objective function";
  }
  identity of-minimize-load-path {
    base objective-function-type;
    description
      "Minimize the load on path(s) objective
      function";
  }
  identity of-maximize-residual-bandwidth {
    base objective-function-type;
    description
      "Maximize the residual bandwidth objective
      function";
  }
  identity of-minimize-agg-bandwidth-consumption {
    base objective-function-type;
    description
      "minimize the aggregate bandwidth consumption
```

```
        objective function";
    }
    identity of-minimize-load-most-loaded-link {
        base objective-function-type;
        description
            "Minimize the load on the most loaded link
             objective function";
    }
    identity of-minimize-cost-path-set {
        base objective-function-type;
        description
            "Minimize the cost on a path set objective
             function";
    }
}

identity path-computation-method {
    description
        "base identity for supported path computation
         mechanisms";
}

identity path-locally-computed {
    base path-computation-method;
    description
        "indicates a constrained-path LSP in which the
         path is computed by the local LER";
}

identity path-externally-queried {
    base path-computation-method;
    description
        "Constrained-path LSP in which the path is
         obtained by querying an external source, such as a PCE server.
         In the case that an LSP is defined to be externally queried, it
         may also have associated explicit definitions (provided
         to the external source to aid computation); and the path that is
         returned by the external source is not required to provide a
         wholly resolved path back to the originating system - that is to
         say, some local computation may also be required";
}

identity path-explicitly-defined {
    base path-computation-method;
    description
        "constrained-path LSP in which the path is
         explicitly specified as a collection of strict or/and loose
         hops";
}
```

```

/**
 * Typedefs
 */

typedef te-bandwidth {
  type string {
    pattern
      '0[xX](0((\.0?)?[pP](\+)?0?|(\.0?))|'
      + '1(\.([\da-fA-F]{0,5}[02468aAcCeE]?))?[pP](\+)?(12[0-7]|'
      + '1[01]\d|0?\d?\d)?|0[xX][\da-fA-F]{1,8}|\d+'
      + '(',0[xX](0((\.0?)?[pP](\+)?0?|(\.0?))|'
      + '1(\.([\da-fA-F]{0,5}[02468aAcCeE]?))?[pP](\+)?(12[0-7]|'
      + '1[01]\d|0?\d?\d)?|0[xX][\da-fA-F]{1,8}|\d+))*';
  }
  description
    "This is the generic bandwidth type that is a string containing
    a list of numbers separated by commas, with each of these
    number can be non-negative decimal, hex integer, or hex float:
    (dec | hex | float)*(','(dec | hex | float))]
    For packet switching type, a float number is used, such as
    0xlp10.
    For OTN switching type, a list of integers can be used, such
    as '0,2,3,1', indicating 2 odu0's and 1 odu3.
    For DWDM, a list of pairs of slot number and width can be
    used, such as '0, 2, 3, 3', indicating a frequency slot 0 with
    slot width 2 and a frequency slot 3 with slot width 3.";
} // te-bandwidth

typedef te-ds-class {
  type uint8 {
    range "0..7";
  }
  description
    "The Differentiated Class-Type of traffic.";
  reference "RFC4124: section-4.3.1";
}

typedef te-link-direction {
  type enumeration {
    enum INCOMING {
      description
        "explicit route represents an incoming link on a node";
    }
    enum OUTGOING {
      description
        "explicit route represents an outgoing link on a node";
    }
  }
}

```



```
    description
      "enumerated type for specifying direction of link on a node";
  }

  typedef te-label-direction {
    type enumeration {
      enum FORWARD {
        description
          "Label allocated for the forward LSP direction";
      }
      enum REVERSE {
        description
          "Label allocated for the reverse LSP direction";
      }
    }
  }
  description
    "enumerated type for specifying the forward or reverse
    label";
}

typedef te-hop-type {
  type enumeration {
    enum LOOSE {
      description
        "loose hop in an explicit path";
    }
    enum STRICT {
      description
        "strict hop in an explicit path";
    }
  }
  description
    "enumerated type for specifying loose or strict
    paths";
  reference "RFC3209: section-4.3.2";
}

identity LSP_METRIC_TYPE {
  description
    "Base identity for types of LSP metric specification";
}

identity LSP_METRIC_RELATIVE {
  base LSP_METRIC_TYPE;
  description
    "The metric specified for the LSPs to which this identity refers
    is specified as a relative value to the IGP metric cost to the
    LSP's tail-end.";
```

```
}

identity LSP_METRIC_ABSOLUTE {
  base LSP_METRIC_TYPE;
  description
    "The metric specified for the LSPs to which this identity refers
    is specified as an absolute value";
}

identity LSP_METRIC_INHERITED {
  base LSP_METRIC_TYPE;
  description
    "The metric for for the LSPs to which this identity refers is
    not specified explicitly - but rather inherited from the IGP
    cost directly";
}

identity tunnel-type {
  description
    "Base identity from which specific tunnel types are
    derived.";
}

identity tunnel-p2p {
  base tunnel-type;
  description
    "TE point-to-point tunnel type.";
}

identity tunnel-p2mp {
  base tunnel-type;
  description
    "TE point-to-multipoint tunnel type.";
  reference "RFC4875";
}

identity tunnel-action-type {
  description
    "Base identity from which specific tunnel action types
    are derived.";
}

identity tunnel-action-resetup {
  base tunnel-action-type;
  description
    "TE tunnel action resetup. Tears the
    tunnel's current LSP (if any) and
    attempts to re-establish a new LSP";
}
```

```
}

identity tunnel-action-reoptimize {
  base tunnel-action-type;
  description
    "TE tunnel action reoptimize.
     Reoptimizes placement of the tunnel LSP(s)";
}

identity tunnel-action-switchpath {
  base tunnel-action-type;
  description
    "TE tunnel action reoptimize
     Switches the tunnel's LSP to use the specified path";
}

identity te-action-result {
  description
    "Base identity from which specific TE action results
     are derived.";
}

identity te-action-success {
  base te-action-result;
  description "TE action successul.";
}

identity te-action-fail {
  base te-action-result;
  description "TE action failed.";
}

identity tunnel-action-inprogress {
  base te-action-result;
  description "TE action inprogress.";
}

identity tunnel-admin-state-type {
  description
    "Base identity for TE tunnel admin states";
}

identity tunnel-admin-state-up {
  base tunnel-admin-state-type;
  description "Tunnel administratively state up";
}

identity tunnel-admin-state-down {
```

```
    base tunnel-admin-state-type;
    description "Tunnel administratively state down";
}

identity tunnel-state-type {
    description
        "Base identity for TE tunnel states";
}

identity tunnel-state-up {
    base tunnel-state-type;
    description "Tunnel state up";
}

identity tunnel-state-down {
    base tunnel-state-type;
    description "Tunnel state down";
}

identity lsp-state-type {
    description
        "Base identity for TE LSP states";
}

identity lsp-path-computing {
    base lsp-state-type;
    description
        "State path compute in progress";
}

identity lsp-path-computation-ok {
    base lsp-state-type;
    description
        "State path compute successful";
}

identity lsp-path-computatione-failed {
    base lsp-state-type;
    description
        "State path compute failed";
}

identity lsp-state-setting-up {
    base lsp-state-type;
    description
        "State setting up";
}
```

```
identity lsp-state-setup-ok {
  base lsp-state-type;
  description
    "State setup successful";
}

identity lsp-state-setup-failed {
  base lsp-state-type;
  description
    "State setup failed";
}

identity lsp-state-up {
  base lsp-state-type;
  description "State up";
}

identity lsp-state-tearing-down {
  base lsp-state-type;
  description
    "State tearing down";
}

identity lsp-state-down {
  base lsp-state-type;
  description "State down";
}

identity path-invalidation-action-type {
  description
    "Base identity for TE path invalidation action types";
}

identity path-invalidation-action-drop-type {
  base path-invalidation-action-type;
  description
    "TE path invalidation action drop";
}

identity path-invalidation-action-drop-tear {
  base path-invalidation-action-type;
  description
    "TE path invalidation action tear";
}

identity lsp-restoration-type {
  description
    "Base identity from which LSP restoration types are
```

```
        derived.";
    }

    identity lsp-restoration-restore-any {
        base lsp-restoration-type;
        description
            "Restores when any of the LSPs is affected by a failure";
    }

    identity lsp-restoration-restore-all {
        base lsp-restoration-type;
        description
            "Restores when all the tunnel LSPs are affected by failure";
    }

    identity restoration-scheme-type {
        description
            "Base identity for LSP restoration schemes";
        reference "RFC4872";
    }

    identity restoration-scheme-preconfigured {
        base restoration-scheme-type;
        description
            "Restoration LSP is preconfigured prior to the failure";
    }

    identity restoration-scheme-precomputed {
        base restoration-scheme-type;
        description
            "Restoration LSP is precomputed prior to the failure";
    }

    identity restoration-scheme-presignaled {
        base restoration-scheme-type;
        description
            "Restoration LSP is presignaled prior to the failure";
    }

    identity lsp-protection-type {
        description
            "Base identity from which LSP protection types are
            derived.";
    }

    identity lsp-protection-unprotected {
        base lsp-protection-type;
        description
```

```
    "LSP protection 'Unprotected'";
    reference "RFC4872";
}

identity lsp-protection-reroute-extra {
    base lsp-protection-type;
    description
        "LSP protection '(Full) Rerouting'";
    reference "RFC4872";
}

identity lsp-protection-reroute {
    base lsp-protection-type;
    description
        "LSP protection 'Rerouting without Extra-Traffic'";
    reference "RFC4872";
}

identity lsp-protection-1-for-n {
    base lsp-protection-type;
    description
        "LSP protection '1:N Protection with Extra-Traffic'";
    reference "RFC4872";
}

identity lsp-protection-unidir-1-to-1 {
    base lsp-protection-type;
    description
        "LSP protection '1+1 Unidirectional Protection'";
    reference "RFC4872";
}

identity lsp-protection-bidir-1-to-1 {
    base lsp-protection-type;
    description
        "LSP protection '1+1 Bidirectional Protection'";
    reference "RFC4872";
}

identity lsp-protection-extra-traffic {
    base lsp-protection-type;
    description
        "LSP protection 'Extra-Traffic'";
    reference
        "ITU-T G.808, RFC 4427.";
}

identity lsp-protection-state {
```

```
    description
      "Base identity of protection states for reporting
      purposes.";
  }

  identity normal {
    base lsp-protection-state;
    description "Normal state.";
  }

  identity signal-fail-of-protection {
    base lsp-protection-state;
    description
      "There is a SF condition on the protection transport
      entity which has higher priority than the FS command.";
    reference
      "ITU-T G.873.1, G.8031, G.8131";
  }

  identity lockout-of-protection {
    base lsp-protection-state;
    description
      "A Loss of Protection (LoP) command is active.";
    reference
      "ITU-T G.808, RFC 4427";
  }

  identity forced-switch {
    base lsp-protection-state;
    description
      "A forced switch (FS) command is active.";
    reference
      "ITU-T G.808, RFC 4427";
  }

  identity signal-fail {
    base lsp-protection-state;
    description
      "There is a SF condition on either the working
      or the protection path.";
    reference
      "ITU-T G.808, RFC 4427";
  }

  identity signal-degrade {
    base lsp-protection-state;
    description
      "There is an SD condition on either the working or the
```



```
        protection path.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity manual-switch {
    base lsp-protection-state;
    description
        "A manual switch (MS) command is active.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity wait-to-restore {
    base lsp-protection-state;
    description
        "A wait time to restore (WTR) is running.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity do-not-revert {
    base lsp-protection-state;
    description
        "A DNR condition is active because of a non-revertive
        behavior.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity failure-of-protocol {
    base lsp-protection-state;
    description
        "The protection is not working because of a failure of
        protocol condition.";
    reference
        "ITU-T G.873.1, G.8031, G.8131";
}

identity protection-external-commands {
    description
        "Protection external commands for trouble shooting
        purposes.";
}

identity action-freeze {
    base protection-external-commands;
    description
```

```
        "A temporary configuration action initiated by an operator
        command to prevent any switch action to be taken and as such
        freezes the current state.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity clear-freeze {
    base protection-external-commands;
    description
        "An action that clears the active freeze state.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity action-lockout-of-normal {
    base protection-external-commands;
    description
        "A temporary configuration action initiated by an operator
        command to ensure that the normal traffic is not allowed
        to use the protection transport entity.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity clear-lockout-of-normal {
    base protection-external-commands;
    description
        "An action that clears the active lockout of normal state.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity action-lockout-of-protection {
    base protection-external-commands;
    description
        "A temporary configuration action initiated by an operator
        command to ensure that the protection transport entity is
        temporarily not available to transport a traffic signal
        (either normal or extra traffic).";
    reference
        "ITU-T G.808, RFC 4427";
}

identity action-forced-switch {
    base protection-external-commands;
    description
        "A switch action initiated by an operator command to swith
```

```
        the extra traffic signal, the normal traffic signal, or the
        null signal to the protection transport entity, unless an
        equal or higher priority switch command is in effect.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity action-manual-switch {
    base protection-external-commands;
    description
        "A switch action initiated by an operator command to switch
        the extra traffic signal, the normal traffic signal #i, or
        the null signal to the protection transport entity, unless
        a fault condition exists on other transport entities or an
        equal or higher priority switch command is in effect.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity action-exercise {
    base protection-external-commands;
    description
        "An action to start testing if the APS communication is
        operating correctly. It is lower priority than any other
        state or command.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity clear {
    base protection-external-commands;
    description
        "An action that clears the active near-end lockout of
        protection, forced switch, manual switch, WTR state,
        or exercise command.";
    reference
        "ITU-T G.808, RFC 4427";
}

identity switching-capabilities {
    description
        "Base identity for interface switching capabilities";
    reference "RFC3471";
}

identity switching-pscl {
```

```
    base switching-capabilities;
    description
      "Packet-Switch Capable-1 (PSC-1)";
    reference "RFC3471";
  }

  identity switching-evpl {
    base switching-capabilities;
    description
      "Ethernet Virtual Private Line (EVPL)";
  }

  identity switching-l2sc {
    base switching-capabilities;
    description
      "Layer-2 Switch Capable (L2SC)";
    reference "RFC3471";
  }

  identity switching-tdm {
    base switching-capabilities;
    description
      "Time-Division-Multiplex Capable (TDM)";
    reference "RFC3471";
  }

  identity switching-otn {
    base switching-capabilities;
    description
      "OTN-TDM capable";
  }

  identity switching-dcsc {
    base switching-capabilities;
    description
      "Data Channel Switching Capable (DCSC)";
  }

  identity switching-lsc {
    base switching-capabilities;
    description
      "Lambda-Switch Capable (LSC)";
    reference "RFC3471";
  }

  identity switching-fsc {
    base switching-capabilities;
    description
```

```
        "Fiber-Switch Capable (FSC)";
    reference "RFC3471";
}

identity lsp-encoding-types {
    description
        "Base identity for encoding types";
    reference "RFC3471";
}

identity lsp-encoding-packet {
    base lsp-encoding-types;
    description
        "Packet LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-ethernet {
    base lsp-encoding-types;
    description
        "Ethernet LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-pdh {
    base lsp-encoding-types;
    description
        "ANSI/ETSI LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-sdh {
    base lsp-encoding-types;
    description
        "SDH ITU-T G.707 / SONET ANSI T1.105 LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-digital-wrapper {
    base lsp-encoding-types;
    description
        "Digital Wrapper LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-lambda {
    base lsp-encoding-types;
    description
```

```
        "Lambda (photonic) LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-fiber {
    base lsp-encoding-types;
    description
        "Fiber LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-fiber-channel {
    base lsp-encoding-types;
    description
        "FiberChannel LSP encoding";
    reference "RFC3471";
}

identity lsp-encoding-oduk {
    base lsp-encoding-types;
    description
        "G.709 ODUk (Digital Path)LSP encoding";
}

identity lsp-encoding-optical-channel {
    base lsp-encoding-types;
    description
        "Line (e.g., 8B/10B) LSP encoding";
}

identity lsp-encoding-line {
    base lsp-encoding-types;
    description
        "Line (e.g., 8B/10B) LSP encoding";
}

identity path-signaling-type {
    description
        "base identity from which specific LSPs path
        setup types are derived";
}

identity path-setup-static {
    base path-signaling-type;
    description
        "Static LSP provisioning path setup";
}
```

```
identity path-setup-rsvp {
  base path-signaling-type;
  description
    "RSVP-TE signaling path setup";
  reference "RFC3209";
}

identity path-setup-sr {
  base path-signaling-type;
  description
    "Segment-routing path setup";
}

identity path-scope-type {
  description
    "base identity from which specific path
    scope types are derived";
}

identity path-scope-segment {
  base path-scope-type;
  description
    "Path scope segment";
}

identity path-scope-end-to-end {
  base path-scope-type;
  description
    "Path scope end to end";
}

/* TE basic features */
feature p2mp-te {
  description
    "Indicates support for P2MP-TE";
  reference "RFC4875";
}

feature frr-te {
  description
    "Indicates support for TE FastReroute (FRR)";
  reference "RFC4090";
}

feature extended-admin-groups {
  description
    "Indicates support for TE link extended admin
    groups.";
```

```
    reference "RFC7308";
  }

  feature named-path-affinities {
    description
      "Indicates support for named path affinities";
  }

  feature named-extended-admin-groups {
    description
      "Indicates support for named extended admin groups";
  }

  feature named-srlg-groups {
    description
      "Indicates support for named SRLG groups";
  }

  feature named-path-constraints {
    description
      "Indicates support for named path constraints";
  }

  feature path-optimization-metric {
    description
      "Indicates support for path optimization metric";
  }

  feature path-optimization-objective-function {
    description
      "Indicates support for path optimization objective function";
  }

  identity route-usage-type {
    description
      "Base identity for route usage";
  }

  identity route-include-ero {
    base route-usage-type;
    description
      "Include ERO from route";
  }

  identity route-exclude-ero {
    base route-usage-type;
    description
      "Exclude ERO from route";
  }
```



```
}

identity route-exclude-srlg {
  base route-usage-type;
  description
    "Exclude SRLG from route";
}

identity path-metric-type {
  description
    "Base identity for path metric type";
}

identity path-metric-te {
  base path-metric-type;
  description
    "TE path metric";
  reference "RFC3785";
}

identity path-metric-igp {
  base path-metric-type;
  description
    "IGP path metric";
  reference "RFC3785";
}

identity path-metric-hop {
  base path-metric-type;
  description
    "Hop path metric";
}

identity path-metric-delay-average {
  base path-metric-type;
  description
    "Unidirectional average link delay";
  reference "RFC7471";
}

identity path-metric-residual-bandwidth {
  base path-metric-type;
  description
    "Unidirectional Residual Bandwidth, which is defined to be
    Maximum Bandwidth [RFC3630] minus the bandwidth currently
    allocated to LSPs.";
  reference "RFC7471";
}
```

```
identity path-metric-optimize-includes {
  base path-metric-type;
  description
    "A metric that optimizes the number of included resources
    specified in a set";
}

identity path-metric-optimize-excludes {
  base path-metric-type;
  description
    "A metric that optimizes the number of excluded resources
    specified in a set";
}

identity path-tiebreaker-type {
  description
    "Base identity for path tie-breaker type";
}

identity path-tiebreaker-minfill {
  base path-tiebreaker-type;
  description
    "Min-Fill LSP path placement";
}

identity path-tiebreaker-maxfill {
  base path-tiebreaker-type;
  description
    "Max-Fill LSP path placement";
}

identity path-tiebreaker-random {
  base path-tiebreaker-type;
  description
    "Random LSP path placement";
}

identity bidir-provisioning-mode {
  description
    "Base identity for bidirectional provisioning
    mode.";
  reference "RFC7551";
}

identity bidir-provisioning-single-sided {
  base bidir-provisioning-mode;
  description
    "Single-sided bidirectional provisioning mode";
}
```

```
    reference "RFC7551";
  }

  identity bidir-provisioning-double-sided {
    base bidir-provisioning-mode;
    description
      "Double-sided bidirectional provisioning mode";
    reference "RFC7551";
  }

  identity bidir-association-type {
    description
      "Base identity for bidirectional association type";
    reference "RFC7551";
  }

  identity bidir-assoc-corouted {
    base bidir-association-type;
    description
      "Co-routed bidirectional association type";
    reference "RFC7551";
  }

  identity bidir-assoc-non-corouted {
    base bidir-association-type;
    description
      "Non co-routed bidirectional association type";
    reference "RFC7551";
  }

  identity resource-affinities-type {
    description
      "Base identity for resource affinities";
    reference "RFC2702";
  }

  identity resource-aff-include-all {
    base resource-affinities-type;
    description
      "The set of attribute filters associated with a
       tunnel all of which must be present for a link
       to be acceptable";
    reference "RFC2702 and RFC3209";
  }

  identity resource-aff-include-any {
    base resource-affinities-type;
    description
```

```
        "The set of attribute filters associated with a
        tunnel any of which must be present for a link
        to be acceptable";
    reference "RFC2702 and RFC3209";
}

identity resource-aff-exclude-any {
    base resource-affinities-type;
    description
        "The set of attribute filters associated with a
        tunnel any of which renders a link unacceptable";
    reference "RFC2702 and RFC3209";
}

typedef optimization-goal {
    type enumeration {
        enum minimize {
            description "Pick lowest path metric goal";
        }
        enum maximize {
            description "Pick highest path metric goal";
        }
        enum randomize {
            description
                "Pick a path at random from list of
                equally favorable ones";
        }
    }
    description "TE optimization goal";
}

identity te-optimization-criterion {
    description
        "Base identity for TE optimization criterion.";
    reference
        "RFC3272: Overview and Principles of Internet Traffic
        Engineering.";
}

identity not-optimized {
    base te-optimization-criterion;
    description "Optimization is not applied.";
}

identity cost {
    base te-optimization-criterion;
    description "Optimized on cost.";
}
```

```
identity delay {
  base te-optimization-criterion;
  description "Optimized on delay.";
}

/*
 * Typedefs
 */

typedef percentage {
  type uint8 {
    range "0..100";
  }
  description
    "Integer indicating a percentage value";
}

typedef performance-metric-normality {
  type enumeration {
    enum "unknown" {
      value 0;
      description
        "Unknown.";
    }
    enum "normal" {
      value 1;
      description
        "Normal.";
    }
    enum "abnormal" {
      value 2;
      description
        "Abnormal. The anomalous bit is set.";
    }
  }
  description
    "Indicates whether a performance metric is normal, abnormal, or
    unknown.";
  reference
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
    RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
    RFC7823: Performance-Based Path Selection for Explicitly
    Routed Label Switched Paths (LSPs) Using TE Metric
    Extensions";
}

typedef te-admin-status {
  type enumeration {
```

```
    enum up {
        description
            "Enabled.";
    }
    enum down {
        description
            "Disabled.";
    }
    enum testing {
        description
            "In some test mode.";
    }
    enum preparing-maintenance {
        description
            "Resource is disabled in the control plane to prepare for
            graceful shutdown for maintenance purposes.";
        reference
            "RFC5817: Graceful Shutdown in MPLS and Generalized MPLS
            Traffic Engineering Networks";
    }
    enum maintenance {
        description
            "Resource is disabled in the data plane for maintenance
            purposes.";
    }
}
description
    "Defines a type representing the administrative status of
    a TE resource.";
}

typedef te-global-id {
    type uint32;
    description
        "An identifier to uniquely identify an operator, which can be
        either a provider or a client.
        The definition of this type is taken from RFC6370 and RFC5003.
        This attribute type is used solely to provide a globally
        unique context for TE topologies.";
}

typedef te-link-access-type {
    type enumeration {
        enum point-to-point {
            description
                "The link is point-to-point.";
        }
        enum multi-access {
```

```
        description
            "The link is multi-access, including broadcast and NBMA.";
    }
}
description
    "Defines a type representing the access type of a TE link.";
reference
    "RFC3630: Traffic Engineering (TE) Extensions to OSPF
    Version 2.";
}

typedef te-node-id {
    type yang:dotted-quad;
    description
        "An identifier for a node in a topology.
        The identifier is represented as 32-bit unsigned integer in
        the dotted-quad notation.
        This attribute is mapped to Router ID in
        RFC3630, RFC5329, RFC5305, and RFC6119.";
}

typedef te-oper-status {
    type enumeration {
        enum up {
            description
                "Operational up.";
        }
        enum down {
            description
                "Operational down.";
        }
        enum testing {
            description
                "In some test mode.";
        }
        enum unknown {
            description
                "Status cannot be determined for some reason.";
        }
        enum preparing-maintenance {
            description
                "Resource is disabled in the control plane to prepare for
                graceful shutdown for maintenance purposes.";
            reference
                "RFC5817: Graceful Shutdown in MPLS and Generalized MPLS
                Traffic Engineering Networks";
        }
        enum maintenance {
```

```
        description
            "Resource is disabled in the data plane for maintenance
            purposes.";
    }
}
description
    "Defines a type representing the operational status of
    a TE resource.";
}

typedef te-path-disjointness {
    type bits {
        bit node {
            position 0;
            description "Node disjoint.";
        }
        bit link {
            position 1;
            description "Link disjoint.";
        }
        bit srlg {
            position 2;
            description "SRLG (Shared Risk Link Group) disjoint.";
        }
    }
    description
        "Type of the resource disjointness for a TE tunnel path.";
    reference
        "RFC4872: RSVP-TE Extensions in Support of End-to-End
        Generalized Multi-Protocol Label Switching (GMPLS)
        Recovery";
} // te-path-disjointness

typedef te-recovery-status {
    type enumeration {
        enum normal {
            description
                "Both the recovery and working spans are fully
                allocated and active, data traffic is being
                transported over (or selected from) the working
                span, and no trigger events are reported.";
        }
        enum recovery-started {
            description
                "The recovery action has been started, but not completed.";
        }
        enum recovery-succeeded {
            description

```



```
        "The recovery action has succeeded. The working span has
        reported a failure/degrade condition and the user traffic
        is being transported (or selected) on the recovery span.";
    }
    enum recovery-failed {
        description
            "The recovery action has failed.";
    }
    enum reversion-started {
        description
            "The reversion has started.";
    }
    enum reversion-failed {
        description
            "The reversion has failed.";
    }
    enum recovery-unavailable {
        description
            "The recovery is unavailable -- either as a result of an
            operator Lockout command or a failure condition detected
            on the recovery span.";
    }
    enum recovery-admin {
        description
            "The operator has issued a command switching the user
            traffic to the recovery span.";
    }
    enum wait-to-restore {
        description
            "The recovery domain is recovering from a failuer/degrade
            condition on the working span that is being controlled by
            the Wait-to-Restore (WTR) timer.";
    }
}
description
    "Defines the status of a recovery action.";
reference
    "RFC4427: Recovery (Protection and Restoration) Terminology
    for Generalized Multi-Protocol Label Switching (GMPLS).
    RFC6378: MPLS Transport Profile (MPLS-TP) Linear Protection";
}

typedef te-template-name {
    type string {
        pattern '/?([a-zA-Z0-9\-\_\.]+)(/[a-zA-Z0-9\-\_\.]+)*';
    }
}
description
    "A type for the name of a TE node template or TE link
```

```
        template.";
    }

typedef te-topology-event-type {
    type enumeration {
        enum "add" {
            value 0;
            description
                "A TE node or te-link has been added.";
        }
        enum "remove" {
            value 1;
            description
                "A TE node or te-link has been removed.";
        }
        enum "update" {
            value 2;
            description
                "A TE node or te-link has been updated.";
        }
    }
    description "TE Event type for notifications";
} // te-topology-event-type

typedef te-topology-id {
    type string {
        pattern
            '([a-zA-Z0-9\-\_\.]+:)*'
            + '/?([a-zA-Z0-9\-\_\.]+)(/[a-zA-Z0-9\-\_\.]+)*';
    }
    description
        "An identifier for a topology.
        It is optional to have one or more prefixes at the begining,
        separated by colons. The prefixes can be the network-types,
        defined in ietf-network.yang, to help user to understand the
        topology better before further inquiry.";
}

typedef te-tp-id {
    type union {
        type uint32; // Unnumbered
        type inet:ip-address; // IPv4 or IPv6 address
    }
    description
        "An identifier for a TE link endpoint on a node.
        This attribute is mapped to local or remote link identifier in
        RFC3630 and RFC5305.";
}
```

```
typedef admin-group {
  type binary {
    length 4;
  }
  description
    "Administrative group/Resource class/Color.";
  reference "RFC3630 and RFC5305";
}

typedef extended-admin-group {
  type binary;
  description
    "Extended administrative group/Resource class/Color.";
  reference "RFC7308";
}

typedef admin-groups {
  type union {
    type admin-group;
    type extended-admin-group;
  }
  description "TE administrative group derived type";
}

typedef srlg {
  type uint32;
  description "SRLG type";
  reference "RFC4203 and RFC5307";
}

identity path-computation-srlg-type {
  description
    "Base identity for SRLG path computation";
}

identity srlg-ignore {
  base path-computation-srlg-type;
  description
    "Ignores SRLGs in path computation";
}

identity srlg-strict {
  base path-computation-srlg-type;
  description
    "Include strict SRLG check in path computation";
}

identity srlg-preferred {
```

```
    base path-computation-srlg-type;
    description
        "Include preferred SRLG check in path computation";
}

identity srlg-weighted {
    base path-computation-srlg-type;
    description
        "Include weighted SRLG check in path computation";
}

typedef te-metric {
    type uint32;
    description
        "TE link metric";
    reference "RFC3785";
}

/**
 * TE bandwidth groupings
 */
identity otn-rate-type {
    description
        "Base type to identify OTN bit rates of various information
        structures.";
    reference "RFC7139";
}
identity odu0 {
    base otn-rate-type;
    description
        "ODU0 bit rate.";
}
identity odu1 {
    base otn-rate-type;
    description
        "ODU1 bit rate.";
}
identity odu2 {
    base otn-rate-type;
    description
        "ODU2 bit rate.";
}
identity odu3 {
    base otn-rate-type;
    description
        "ODU3 bit rate.";
}
identity odu4 {
```

```
    base otn-rate-type;
    description
        "ODU4 bit rate.";
}
identity odu2e {
    base otn-rate-type;
    description
        "ODU2e bit rate.";
}
identity oduc {
    base otn-rate-type;
    description
        "ODUCn bit rate.";
}
identity oduflex {
    base otn-rate-type;
    description
        "ODUflex bit rate.";
}

identity wdm-spectrum-type {
    description
        "Base type to identify WDM spectrum type.";
}
identity cwdm {
    base wdm-spectrum-type;
    description "CWDM.";
    reference "RFC6205";
}
identity dwdm {
    base wdm-spectrum-type;
    description "DWDM.";
    reference "RFC6205";
}
identity flexible-grid {
    base wdm-spectrum-type;
    description "Flexible grid.";
    reference "RFC6205";
}

grouping te-bandwidth {
    description
        "This grouping defines the generic TE bandwidth.
        For some known data plane technologies, specific modeling
        structures are specified. The string encoded te-bandwidth
        type is used for un-specified technologies.
        The modeling structure can be augmented later for other
        technologies.";
```

```
    container te-bandwidth {
      description
        "Container that specifies TE bandwidth.";
      choice technology {
        default generic;
        description
          "Data plane technology type.";
        case generic {
          leaf generic {
            type te-bandwidth;
            description
              "Bandwidth specified in a generic format.";
          }
        }
      }
    }
  }
}

/**
 * TE label groupings
 **/
grouping te-label {
  description
    "This grouping defines the generic TE label.
    The modeling structure can be augmented for each technology.
    For un-specified technologies, rt-types:generalized-label
    is used.";
  container te-label {
    description
      "Container that specifies TE label.";
    choice technology {
      default generic;
      description
        "Data plane technology type.";
      case generic {
        leaf generic {
          type rt-types:generalized-label;
          description
            "TE label specified in a generic format.";
        }
      }
    }
  }
  leaf direction {
    type te-label-direction;
    description "Label direction";
  }
}
}
```

```
/**
 * TE performance metric groupings
 **/
grouping performance-metric-container {
  description
    "A container containing performance metric attributes.";
  container performance-metric {
    description
      "Link performance information in real time.";
    reference
      "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
       RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
       RFC7823: Performance-Based Path Selection for Explicitly
       Routed Label Switched Paths (LSPs) Using TE Metric
       Extensions";
    container measurement {
      description
        "Measured performance metric values. Static configuration
         and manual overrides of these measurements are also
         allowed.";
      uses performance-metric-attributes;
    }
    container normality
    {
      description
        "Performance metric normality values.";
      uses performance-metric-normality-attributes;
    }
    uses performance-metric-throttle-container;
  }
} // performance-metric-container

grouping te-topology-identifier {
  description
    "Augmentation for TE topology.";
  container te-topology-identifier {
    description "TE topology identifier container";
    leaf provider-id {
      type te-types:te-global-id;
      description
        "An identifier to uniquely identify a provider.";
    }
    leaf client-id {
      type te-types:te-global-id;
      description
        "An identifier to uniquely identify a client.";
    }
    leaf topology-id {
```

```
    type te-types:te-topology-id;
    description
      "It is presumed that a datastore will contain many
       topologies. To distinguish between topologies it is
       vital to have UNIQUE topology identifiers.";
  }
}

grouping performance-metric-attributes {
  description
    "Link performance information in real time.";
  reference
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
     RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
     RFC7823: Performance-Based Path Selection for Explicitly
     Routed Label Switched Paths (LSPs) Using TE Metric
     Extensions";
  leaf unidirectional-delay {
    type uint32 {
      range 0..16777215;
    }
    description "Delay or latency in micro seconds.";
  }
  leaf unidirectional-min-delay {
    type uint32 {
      range 0..16777215;
    }
    description "Minimum delay or latency in micro seconds.";
  }
  leaf unidirectional-max-delay {
    type uint32 {
      range 0..16777215;
    }
    description "Maximum delay or latency in micro seconds.";
  }
  leaf unidirectional-delay-variation {
    type uint32 {
      range 0..16777215;
    }
    description "Delay variation in micro seconds.";
  }
  leaf unidirectional-packet-loss {
    type decimal64 {
      fraction-digits 6;
      range "0 .. 50.331642";
    }
  }
}
```



```
    }
    description
      "Packet loss as a percentage of the total traffic sent
      over a configurable interval. The finest precision is
      0.000003%.";
  }
  leaf unidirectional-residual-bandwidth {
    type rt-types:bandwidth-ieee-float32;
    description
      "Residual bandwidth that subtracts tunnel
      reservations from Maximum Bandwidth (or link capacity)
      [RFC3630] and provides an aggregated remainder across QoS
      classes.";
  }
  leaf unidirectional-available-bandwidth {
    type rt-types:bandwidth-ieee-float32;
    description
      "Available bandwidth that is defined to be residual
      bandwidth minus the measured bandwidth used for the
      actual forwarding of non-RSVP-TE LSP packets. For a
      bundled link, available bandwidth is defined to be the
      sum of the component link available bandwidths.";
  }
  leaf unidirectional-utilized-bandwidth {
    type rt-types:bandwidth-ieee-float32;
    description
      "Bandwidth utilization that represents the actual
      utilization of the link (i.e. as measured in the router).
      For a bundled link, bandwidth utilization is defined to
      be the sum of the component link bandwidth
      utilizations.";
  }
} // performance-metric-attributes

grouping performance-metric-normality-attributes {
  description
    "Link performance metric normality attributes.";
  reference
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
    RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
    RFC7823: Performance-Based Path Selection for Explicitly
    Routed Label Switched Paths (LSPs) Using TE Metric
    Extensions";
  leaf unidirectional-delay {
    type te-types:performance-metric-normality;
    description "Delay normality.";
  }
  leaf unidirectional-min-delay {
```

```
    type te-types:performance-metric-normality;
    description "Minimum delay or latency normality.";
  }
  leaf unidirectional-max-delay {
    type te-types:performance-metric-normality;
    description "Maximum delay or latency normality.";
  }
  leaf unidirectional-delay-variation {
    type te-types:performance-metric-normality;
    description "Delay variation normality.";
  }
  leaf unidirectional-packet-loss {
    type te-types:performance-metric-normality;
    description "Packet loss normality.";
  }
  leaf unidirectional-residual-bandwidth {
    type te-types:performance-metric-normality;
    description "Residual bandwidth normality.";
  }
  leaf unidirectional-available-bandwidth {
    type te-types:performance-metric-normality;
    description "Available bandwidth normality.";
  }
  leaf unidirectional-utilized-bandwidth {
    type te-types:performance-metric-normality;
    description "Bandwidth utilization normality.";
  }
} // performance-metric-normality-attributes

grouping performance-metric-throttle-container {
  description
    "A container controlling performance metric throttle.";
  container throttle {
    must "suppression-interval >= measure-interval" {
      error-message
        "suppression-interval cannot be less than
        measure-interval.";
      description
        "Constraint on suppression-interval and
        measure-interval.";
    }
  }
  description
    "Link performance information in real time.";
  reference
    "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.
    RFC7810: IS-IS Traffic Engineering (TE) Metric Extensions.
    RFC7823: Performance-Based Path Selection for Explicitly
    Routed Label Switched Paths (LSPs) Using TE Metric
```

```
    Extensions";
  leaf unidirectional-delay-offset {
    type uint32 {
      range 0..16777215;
    }
    description
      "Offset value to be added to the measured delay value.";
  }
  leaf measure-interval {
    type uint32;
    default 30;
    description
      "Interval in seconds to measure the extended metric
        values.";
  }
  leaf advertisement-interval {
    type uint32;
    description
      "Interval in seconds to advertise the extended metric
        values.";
  }
  leaf suppression-interval {
    type uint32 {
      range "1 .. max";
    }
    default 120;
    description
      "Interval in seconds to suppress advertising the extended
        metric values.";
  }
  container threshold-out {
    uses performance-metric-attributes;
    description
      "If the measured parameter falls outside an upper bound
        for all but the min delay metric (or lower bound for
        min-delay metric only) and the advertised value is not
        already outside that bound, anomalous announcement will be
        triggered.";
  }
  container threshold-in {
    uses performance-metric-attributes;
    description
      "If the measured parameter falls inside an upper bound
        for all but the min delay metric (or lower bound for
        min-delay metric only) and the advertised value is not
        already inside that bound, normal (anomalous-flag cleared)
        announcement will be triggered.";
  }
}
```

```
    container threshold-accelerated-advertisement {
      description
        "When the difference between the last advertised value and
        current measured value exceed this threshold, anomalous
        announcement will be triggered.";
      uses performance-metric-attributes;
    }
  }
} // performance-metric-throttle-container

/**
 * TE tunnel generic groupings
 **/

/* Tunnel path selection parameters */
grouping explicit-route-hop {
  description
    "The explicit route subobject grouping";
  leaf index {
    type uint32;
    description "ERO subobject index";
  }
  choice type {
    description
      "The explicit route subobject type";
    case num-unnum-hop {
      container num-unnum-hop {
        leaf node-id {
          type te-types:te-node-id;
          description
            "The identifier of a node in the TE topology.";
        }
        leaf link-tp-id {
          type te-types:te-tp-id;
          description
            "TE link termination point identifier. The combination
            of TE link ID and the TE node ID is used to identify an
            unnumbered TE link.";
        }
      }
      leaf hop-type {
        type te-hop-type;
        description "strict or loose hop";
      }
      leaf direction {
        type te-link-direction;
        description "Unnumbered Link ERO direction";
      }
    }
  }
  description
```

```

        "Numbered and Unnumbered link/node explicit route
        subobject";
    reference
        "RFC3209: section 4.3 for EXPLICIT_ROUTE in RSVP-TE
        RFC3477: Signalling Unnumbered Links in RSVP-TE";
    }
}
case as-number {
    container as-number-hop {
        leaf as-number {
            type binary {
                length 16;
            }
            description "AS number";
        }
        leaf hop-type {
            type te-hop-type;
            description
                "strict or loose hop";
        }
        description
            "Autonomous System explicit route subobject";
    }
}
case label {
    container label-hop {
        description "Label hop type";
        uses te-label;
    }
    description
        "The Label ERO subobject";
}
}
}

grouping record-route-subobject_state {
    description
        "The record route subobject grouping";
    leaf index {
        type uint32;
        description "RRO subobject index";
    }
    choice type {
        description
            "The record route subobject type";
        case numbered {
            leaf address {
                type te-types:te-tp-id;
            }
        }
    }
}

```

```
        description
            "Numbered link TE termination point address.";
    }
    leaf ip-flags {
        type binary {
            length 8;
        }
        description
            "RRO IP address sub-object flags";
        reference "RFC3209";
    }
}
case unnumbered {
    leaf node-id {
        type te-types:te-node-id;
        description
            "The identifier of a node in the TE topology.";
    }
    leaf link-tp-id {
        type te-types:te-tp-id;
        description
            "TE link termination point identifier, used
            together with te-node-id to identify the
            link termination point";
    }
    description
        "Unnumbered link record route subobject";
    reference
        "RFC3477: Signalling Unnumbered Links in
        RSVP-TE";
}
case label {
    container label-hop {
        description "Label hop type";
        uses te-label;
        leaf label-flags {
            type binary {
                length 8;
            }
            description
                "Label sub-object flags";
            reference "RFC3209";
        }
    }
    description
        "The Label RRO subobject";
}
}
```

```
}

grouping label-restriction-info {
  description "Label set item info";
  leaf restriction {
    type enumeration {
      enum inclusive {
        description "The label or label range is inclusive.";
      }
      enum exclusive {
        description "The label or label range is exclusive.";
      }
    }
    description
      "Whether the list item is inclusive or exclusive.";
  }
  leaf index {
    type uint32;
    description
      "Then index of the label restriction list entry.";
  }
  container label-start {
    description
      "This is the starting label if a label range is specified.
       This is the label value if a single label is specified,
       in which case, attribute 'label-end' is not set.";
    uses te-label;
  }
  container label-end {
    description
      "The ending label if a label range is specified;
       This attribute is not set, If a single label is
       specified.";
    uses te-label;
  }
  leaf range-bitmap {
    type binary;
    description
      "When there are gaps between label-start and label-end,
       this attribute is used to specified the possitions
       of the used labels.";
  }
}

grouping label-set-info {
  description
    "Grouping for List of label restrictions specifying what labels
     may or may not be used on a link connectivity.";
```

```
    container label-restrictions {
      description
        "The label restrictions container";
      list label-restriction {
        key "index";
        description
          "The absence of label-set implies that all labels are
           acceptable; otherwise only restricted labels are
           available.";
        reference
          "RFC7579: General Network Element Constraint Encoding
           for GMPLS-Controlled Networks";
        uses label-restriction-info;
      }
    }
  }
}

/** End of TE tunnel groupings */
grouping optimizations_config {
  description "Optimization metrics configuration grouping";
  leaf metric-type {
    type identityref {
      base te-types:path-metric-type;
    }
    description "TE path metric type";
  }
  leaf weight {
    type uint8;
    description "TE path metric normalization weight";
  }
  container explicit-route-exclude-objects {
    when "../metric-type = " +
      "'te-types:path-metric-optimize-excludes'";
    description
      "Container for the exclude route object list";
    uses path-route-exclude-objects;
  }
  container explicit-route-include-objects {
    when "../metric-type = " +
      "'te-types:path-metric-optimize-includes'";
    description
      "Container for the include route object list";
    uses path-route-include-objects;
  }
}

grouping common-constraints_config {
  description
```



```
    "Common constraints grouping that can be set on
      a constraint set or directly on the tunnel";

  uses te-types:te-bandwidth {
    description
      "A requested bandwidth to use for path computation";
  }

  leaf setup-priority {
    type uint8 {
      range "0..7";
    }
    description
      "TE LSP requested setup priority";
    reference "RFC3209";
  }
  leaf hold-priority {
    type uint8 {
      range "0..7";
    }
    description
      "TE LSP requested hold priority";
    reference "RFC3209";
  }
  leaf signaling-type {
    type identityref {
      base te-types:path-signaling-type;
    }
    description "TE tunnel path signaling type";
  }
}

grouping tunnel-constraints_config {
  description
    "Tunnel constraints grouping that can be set on
      a constraint set or directly on the tunnel";
  uses te-types:te-topology-identifier;
  uses te-types:common-constraints_config;
}

grouping path-metrics-bounds_config {
  description "TE path metric bounds grouping";
  leaf metric-type {
    type identityref {
      base te-types:path-metric-type;
    }
    description "TE path metric type";
  }
}
```

```
    leaf upper-bound {
      type uint64;
      description "Upper bound on end-to-end TE path metric";
    }
  }

  grouping path-objective-function_config {
    description "Optimization metrics configuration grouping";
    leaf objective-function-type {
      type identityref {
        base te-types:objective-function-type;
      }
      description
        "Objective function entry";
    }
  }
}

/**
 * TE interface generic groupings
 */
grouping path-route-objects {
  description
    "List of EROs to be included or excluded when performing
    the path computation.";
  container explicit-route-objects {
    description
      "Container for the exclude route object list";
    list route-object-exclude-always {
      key index;
      description
        "List of explicit route objects to always exclude
        from path computation";
      uses te-types:explicit-route-hop;
    }
    list route-object-include-exclude {
      key index;
      description
        "List of explicit route objects to include or
        exclude in path computation";
      leaf explicit-route-usage {
        type identityref {
          base te-types:route-usage-type;
        }
        description "Explicit-route usage.";
      }
      uses te-types:explicit-route-hop {
        augment "type" {
          case srlg {
```

```

        container srlg {
            description "SRLG container";
            leaf srlg {
                type uint32;
                description "SRLG value";
            }
        }
        description "An SRLG value to be included or excluded";
    }
    description
        "Augmentation to generic explicit route for SRLG
        exclusion";
    }
}
}
}
}
}

```

```

grouping path-route-include-objects {
    description
        "List of EROs to be included when performing
        the path computation.";
    list route-object-include-object {
        key index;
        description
            "List of explicit route objects to be included
            in path computation";
        uses te-types:explicit-route-hop;
    }
}

```

```

grouping path-route-exclude-objects {
    description
        "List of EROs to be included when performing
        the path computation.";
    list route-object-exclude-object {
        key index;
        description
            "List of explicit route objects to be excluded
            in path computation";
        uses te-types:explicit-route-hop {
            augment "type" {
                case srlg {
                    container srlg {
                        description "SRLG container";
                        leaf srlg {
                            type uint32;
                            description "SRLG value";
                        }
                    }
                }
            }
        }
    }
}

```

```
    }
  }
  description "An SRLG value to be included or excluded";
}
description
  "Augmentation to generic explicit route for SRLG exclusion";
}
}
}

grouping generic-path-metric-bounds {
  description "TE path metric bounds grouping";
  container path-metric-bounds {
    description "TE path metric bounds container";
    list path-metric-bound {
      key metric-type;
      description "List of TE path metric bounds";
      uses path-metrics-bounds_config;
    }
  }
}

grouping generic-path-optimization {
  description "TE generic path optimization grouping";

  container optimizations {
    description
      "The objective function container that includes
      attributes to impose when computing a TE path";

    choice algorithm {
      description "Optimizations algorithm.";
      case metric {
        if-feature path-optimization-metric;
        /* Optimize by metric */
        list optimization-metric {
          key "metric-type";
          description "TE path metric type";
          uses optimizations_config;
        }
      }
      /* Tiebreakers */
      container tiebreakers {
        description
          "The list of tiebreaker criterion to apply
          on an equally favored set of paths to pick best";
        list tiebreaker {
          key "tiebreaker-type";

```

```

        description
            "The list of tiebreaker criterion to apply
            on an equally favored set of paths to pick best";
        leaf tiebreaker-type {
            type identityref {
                base te-types:path-metric-type;
            }
            description "The objective function";
        }
    }
}

case objective-function {
    if-feature path-optimization-objective-function;
    /* Objective functions */
    container objective-function {
        description
            "The objective function container that includes
            attributes to impose when computing a TE path";
        uses path-objective-function_config;
    }
}

}

}

}

grouping generic-path-affinities {
    description
        "Path affinities grouping";
    container path-affinities {
        description
            "Path affinities container";
        list constraint {
            key "usage";
            description
                "List of named affinity constraints";
            leaf usage {
                type identityref {
                    base resource-affinities-type;
                }
                description "Affinities usage";
            }
            leaf value {
                type admin-groups;
                description "Affinity value";
            }
        }
    }
}

```

```
}

grouping generic-path-srlgs {
  description
    "Path SRLG grouping";
  container path-srlgs {
    description
      "Path SRLG properties container";
    leaf usage {
      type identityref {
        base te-types:route-exclude-srlg;
      }
      description "SRLG usage";
    }
    leaf-list values {
      type srlg;
      description "SRLG value";
    }
  }
}

grouping generic-path-disjointness {
  description "Path disjointness grouping";
  leaf disjointness {
    type te-types:te-path-disjointness;
    description
      "The type of resource disjointness.
      Under primary path, disjointness level applies to
      all secondary LSPs. Under secondary, disjointness
      level overrides the one under primary";
  }
}

grouping common-path-constraints-attributes {
  description
    "Common path constraints configuration grouping";
  uses common-constraints_config;
  uses generic-path-metric-bounds;
  uses generic-path-affinities;
  uses generic-path-srlgs;
}

grouping generic-path-constraints {
  description
    "Global named path constraints configuration
    grouping";
  container path-constraints {
    description "TE named path constraints container";
  }
}
```

```

        uses common-path-constraints-attributes;
        uses generic-path-disjointness;
    }
}

grouping generic-path-properties {
    description "TE generic path properties grouping";
    container path-properties {
        config false;
        description "The TE path 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 generic-path-affinities;
    uses generic-path-srlgs;
    container path-route-objects {
        description
            "Container for the list of route objects either returned by
            the computation engine or actually used by an LSP";
        list path-route-object {
            key index;
            description
                "List of route objects either returned by the computation
                engine or actually used by an LSP";
            uses explicit-route-hop;
        }
    }
}
}
}
<CODE ENDS>

```

Figure 7: TE basic types YANG module

```

<CODE BEGINS> file "ietf-te@2018-07-01.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;
}

import ietf-inet-types {
  prefix inet;
}

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@cisco.com>

  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@jabil.com>

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

description
  "YANG data module for TE configuration,
```



```
    state, RPC and notifications.";

revision "2018-07-01" {
    description "Latest update to TE generic YANG module.";
    reference "TBA";
}

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

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

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";
}

/**
 * TE tunnel generic groupings
 */
grouping path-affinities-contents_config {
    description
        "Path affinities constraints grouping";
}
```

```
reference "RFC3630 and RFC5305";
leaf usage {
  type identityref {
    base te-types:resource-affinities-type;
  }
  description "Affinities usage";
}
choice style {
  description
    "Path affinities representation style";
  case value {
    leaf value {
      type te-types:admin-groups;
      description
        "Bitmap indicating what bits are of significance";
    }
  }
  case named {
    list affinity-names {
      key "name";
      leaf name {
        type string;
        description "Affinity name";
      }
      description "List of named affinities";
    }
  }
}

grouping path-affinities {
  description "Path affinities grouping";
  reference "RFC 3209";
  container path-affinities {
    description "Path affinities container";
    list constraints {
      key "usage";
      description "List of named affinity constraints";
      uses path-affinities-contents_config;
    }
  }
}

grouping path-srlgs-values_config {
  description "Path SRLG values properties grouping";
  reference "RFC4203 and RFC5307";
  leaf usage {
    type identityref {
```

```

        base te-types:route-exclude-srlg;
    }
    description "SRLG usage";
}
leaf-list values {
    type te-types:srlg;
    description "SRLG value";
    reference "RFC4203 and RFC5307";
}
}

grouping path-srlgs {
    description "Path SRLG properties grouping";
    container path-srlgs {
        description "Path SRLG properties container";
        choice style {
            description "Type of SRLG representation";
            case values {
                uses path-srlgs-values_config;
            }
            case named {
                container constraints {
                    description "SRLG named constraints";
                    list constraint {
                        key "usage";
                        leaf usage {
                            type identityref {
                                base te-types:route-exclude-srlg;
                            }
                            description "SRLG usage";
                        }
                    }
                    container constraint {
                        description "Container for named SRLG list";
                        list srlg-names {
                            key "name";
                            leaf name {
                                type string;
                                description "The SRLG name";
                            }
                        }
                        description "List named SRLGs";
                    }
                }
            }
        }
        description "List of named SRLG constraints";
    }
}
}
}
}
}

```

```
}

grouping p2p-reverse-primary-path-properties {
  description "tunnel path properties.";
  reference "RFC7551";
  container p2p-reverse-primary-path {
    description "Tunnel reverse primary path properties";
    uses p2p-path-reverse-properties_config;
    uses path-constraints-common_config;
    container state {
      config false;
      description
        "Configuration applied parameters and state";
      uses p2p-path-properties_state;
    }
    container p2p-reverse-secondary-path {
      description "Tunnel reverse secondary path properties";
      uses p2p-reverse-path-candidate-secondary-path-config;
    }
  }
}

grouping p2p-secondary-path-properties {
  description "tunnel path properties.";
  uses p2p-path-properties_config;
  uses path-constraints-common_config;
  uses protection-restoration-params_config;
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    uses p2p-path-properties_state;
  }
}

grouping p2p-primary-path-properties {
  description
    "TE tunnel primary path properties grouping";
  uses p2p-path-properties_config;
  uses path-constraints-common_config;
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    uses p2p-path-properties_state;
  }
}
```

```
grouping path-properties_state {
  description "Computed path properties grouping";
  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";
  }
}

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 leafref {
          path "../state/metric-type";
        }
        description "TE path metric type";
      }
    }
    container state {
      config false;
      description
        "Configuration applied parameters and state";
      uses path-properties_state;
    }
  }
  uses path-affinities;
  uses path-srlgs;
  container path-route-objects {
    description
      "Container for the list of computed route objects
      as returned by the computation engine";
    list path-computed-route-object {
      key index;
      description
        "List of computed route objects returned by the
        computation engine";
      leaf index {
        type leafref {
          path "../state/index";
        }
      }
    }
  }
}
```

```
    }
    description "Index of computed route object";
  }
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    uses te-types:explicit-route-hop;
  }
}
}
uses shared-resources-tunnels;
}
}

grouping p2p-path-properties_state {
  description "TE per path state parameters";
  uses path-properties {
    description "The TE path computed properties";
  }
  container lsps {
    description "TE LSPs 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 shared-resources-tunnels_state;
      uses lsp-record-route-information_state;
      uses path-properties {
        description "The TE path actual properties";
      }
    }
  }
}

grouping p2p-path-properties-common_config {
  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;
    }
  }
}
```

```
    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;
    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;
        description "Path scope if segment or an end-to-end path";
    }
}

grouping p2p-path-reverse-properties_config {
    description
        "TE tunnel reverse path properties configuration
        grouping";
    uses p2p-path-properties-common_config;
    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-path-properties_config {
    description
        "TE tunnel path properties configuration grouping";
    uses p2p-path-properties-common_config;
    uses te-types:generic-path-optimization;
    leaf preference {
        type uint8 {
            range "1..255";
        }
        description
            "Specifies a preference for this path. The lower the
            number higher the preference";
    }
    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";
    }
}
```



```
}

/* TE tunnel configuration data */
grouping tunnel-p2mp-params_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 "RFC 3209";
  }
  leaf description {
    type string;
    description
      "Textual description for this TE tunnel";
  }
}

grouping hierarchical-link_config {
  description
    "Hierarchical link configuration grouping";
  reference "RFC4206";
  leaf local-te-node-id {
    type te-types:te-node-id;
    description
      "Local TE node identifier";
  }
  leaf local-te-link-tp-id {
    type te-types:te-tp-id;
    description
      "Local TE link termination point identifier";
  }
  leaf remote-te-node-id {
    type te-types:te-node-id;
    description
      "Remote TE node identifier";
  }
  uses te-types:te-topology-identifier;
}

grouping hierarchical-link {
  description
    "Hierarchical link grouping";
```

```
reference "RFC4206";
container hierarchical-link {
  description
    "Identifies a hierarchical link (in client layer)
    that this tunnel is associated with.";
  uses hierarchical-link_config;
}

grouping protection-restoration-params_state {
  description
    "Protection parameters grouping";
  leaf lockout-of-normal {
    type boolean;
    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
      "ITU-T G.808, RFC 4427";
  }
  leaf freeze {
    type boolean;
    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
      "ITU-T G.808, RFC 4427";
  }
  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";
    }
}
description "LSP role type";
reference "rfc4872, section 4.2.1";
}

leaf lsp-protection-state {
    type identityref {
        base te-types:lsp-protection-state;
    }
    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;
    description
        "Indicates the te-node-id of the protection group
        ingress node when the APS state represents an external
        command (LoP, SF, MS) applied to it or a WTR timer
        running on it. If the external command is not applied to
        the ingress node or the WTR timer is not running on it,
        this attribute is not specified. If 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;
    description
        "Indicates the te-node-id of the protection group egress node
        when the APS state represents an external command (LoP, SF,
        MS) applied to it or a WTR timer running on it. If the
        external command is not applied to the ingress node or
        the WTR timer is not running on it, this attribute is not
        specified. If 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-params_config {
    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;
    }
    description "LSP protection type.";
  }
  leaf protection-reversion-disable {
    type boolean;
    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";
    }
    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
      "ITU-T G.808.1";
  }
}
```

```
    }  
  }  
  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;  
      }  
      description "LSP restoration type.";  
    }  
    leaf restoration-scheme {  
      type identityref {  
        base te-types:restoration-scheme-type;  
      }  
      description "LSP restoration scheme.";  
    }  
    leaf restoration-reversion-disable {  
      type boolean;  
      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_config {
    description
        "Grouping for tunnel dependency list of tunnels";
    container dependency-tunnels {
        description "Dependency tunnels list";
        list dependency-tunnel {
            key "name";
            description "Dependency tunnel entry";
            leaf name {
                type leafref {
                    path "../../../../../tunnels/tunnel/name";
                    require-instance false;
                }
                description "Dependency tunnel name";
            }
            leaf encoding {
                type identityref {
                    base te-types:lsp-encoding-types;
                }
                description "LSP encoding type";
                reference "RFC3945";
            }
            leaf switching-type {
                type identityref {
                    base te-types:switching-capabilities;
                }
                description "LSP switching type";
                reference "RFC3945";
            }
        }
    }
}

grouping tunnel-p2p-params_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;
        description
            "Textual description for this TE tunnel";
    }
    leaf encoding {
        type identityref {
            base te-types:lsp-encoding-types;
        }
        description "LSP encoding type";
        reference "RFC3945";
    }
    leaf switching-type {
        type identityref {
            base te-types:switching-capabilities;
        }
        description "LSP switching type";
        reference "RFC3945";
    }
    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";
        }
        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 binary;
      description
        "TE tunnel source termination point identifier.";
    }
    leaf dst-tp-id {
      type binary;
      description
        "TE tunnel destination termination point identifier.";
    }
    leaf bidirectional {
      type boolean;
      default 'false';
      description "TE tunnel bidirectional";
    }
    uses tunnel-p2p-associations_config;
    uses protection-restoration-params_config;
    uses te-types:tunnel-constraints_config;
    uses p2p-dependency-tunnels_config;
    uses hierarchical-link;
  }

  grouping tunnel-p2p-associations_config {
    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 inet:ip-address;
      description "Association source";
      reference "RFC4872";
    }
    leaf global-source {
      type inet:ip-address;
      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 inet:ip-address;
      description "Association source";
    }
    leaf global-source {
      type inet:ip-address;
      description "Association global source";
      reference "RFC4872";
    }
    leaf extended-ID {
      type binary;
      description "Association extended ID";
      reference "RFC4872";
    }
  }
}

grouping tunnel-p2p-params_state {
  description
    "State parameters relating to TE tunnel";
  leaf operational-state {
```

```
    type identityref {
      base te-types:tunnel-state-type;
    }
    default te-types:tunnel-state-up;
    description "TE tunnel administrative state.";
  }
}

grouping access-segment-info {
  description
    "info related to a segment";
  container forward {
    description
      "for the forward direction of this tunnel";
    uses te-types:label-set-info;
  }
  container reverse {
    description
      "for the reverse direction of this tunnel";
    uses te-types:label-set-info;
  }
}

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 access-segment-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 access-segment-info;
  }
}
```

```
/* TE tunnel configuration/state grouping */
grouping tunnel-p2mp-properties {
  description
    "Top level grouping for P2MP tunnel properties.";
  uses tunnel-p2mp-params_config;
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    leaf operational-state {
      type identityref {
        base te-types:tunnel-state-type;
      }
      default te-types:tunnel-state-up;
      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;
    }
    description
      "Signaling protocol used to set up this tunnel";
  }
}

grouping p2p-reverse-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;
  }
  description
    "Signaling protocol used to set up this tunnel";
}
}

grouping p2p-path-candidate-secondary-path-state {
  description
    "Operational state parameters relating to a secondary path
    which is a candidate for a particular primary path";

  leaf active {
    type boolean;
    description
      "Indicates the current active path option that has
      been selected of the candidate secondary paths";
  }
}

grouping tunnel-p2p-properties {
  description
    "Top level grouping for tunnel properties.";
  uses tunnel-p2p-params_config;
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    uses tunnel-p2p-params_state;
  }
  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-reverse-primary-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;

    container state {
      config false;
      description
        "Configuration applied parameters and state";
      uses p2p-path-candidate-secondary-path-state;
    }
  }
}

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 te:tunnel-ref;
    description
```

```
        "Reference to the tunnel that sharing secondary path
        resources with this tunnel";
    }
}
grouping shared-resources-tunnels {
    description
        "Set of tunnels that share secondary path resources with
        this tunnel";
    container shared-resources-tunnels {
        description
            "Set of tunnels that share secondary path resources with
            this tunnel";
        leaf-list lsp-shared-resources-tunnel {
            type te: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
            "ITU-T G.808, RFC 4427";
    }
    leaf extra-traffic-tunnel-ref {
        type te: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-subobjects {
        description "RSVP recorded route object information";
        list record-route-subobject {
            when "../../origin-type = 'ingress'" {
                description "Applicable on non-ingress LSPs only";
            }
            key "index";
            description "Record route sub-object list";
            uses te-types:record-route-subobject_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_config {
    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_config;
        }
    }
}
```

```
/* Global named admin-srlgs configuration data */
grouping named-srlgs_config {
  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_config;
    }
  }
}

/* Global named paths constraints configuration data */
grouping path-constraints_state {
  description
    "TE path constraints state";
  leaf bandwidth-generic_state {
    type te-types:te-bandwidth;
    description
      "A technology agnostic requested bandwidth to use
       for path computation";
  }
  leaf disjointness_state {
```

```
    type te-types:te-path-disjointness;
    description
      "The type of resource disjointness.";
  }
}

grouping path-constraints-common_config {
  description
    "Global named path constraints configuration
    grouping";
  uses te-types:common-path-constraints-attributes;
  uses te-types:generic-path-disjointness;
  uses te-types:path-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 path-constraints {
  description "Per path constraints";
  uses path-constraints-common_config;
  container state {
    config false;
    description
      "Configuration applied parameters and state";
    uses path-constraints_state;
  }
}

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;
    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 inet:ip-address;
  description
    "Tunnel sender address extracted from
     SENDER_TEMPLATE object";
  reference "RFC3209";
}
leaf destination {
  type inet:ip-address;
  description
    "Tunnel endpoint address extracted from
     SESSION object";
  reference "RFC3209";
}
leaf tunnel-id {
  type uint16;
  description
    "Tunnel identifier used in the SESSION
     that remains constant over the life
     of the tunnel.";
  reference "RFC3209";
}
leaf lsp-id {
  type uint16;
  description
    "Identifier used in the SENDER_TEMPLATE
     and the FILTER_SPEC that can be changed
     to allow a sender to share resources with
     itself.";
  reference "RFC3209";
}
leaf extended-tunnel-id {
  type inet:ip-address;
  description
    "Extended Tunnel ID of the LSP.";
  reference "RFC3209";
}
leaf 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;
```

```
    }
    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";
      }
    }
  }
  description
    "Origin type of LSP relative to the location
    of the local switch in the path.";
}

leaf lsp-resource-status {
  type enumeration {
    enum primary {
      description
        "A primary LSP is a fully established LSP for
        which the resource allocation has been committed
        at the data plane";
    }
    enum secondary {
      description
        "A secondary LSP is an LSP that has been provisioned
        in the control plane only; e.g. resource allocation
        has not been committed at the data plane";
    }
  }
  description "LSP resource allocation type";
  reference "rfc4872, section 4.2.1";
}

  uses protection-restoration-params_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 lsp-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 te:tunnel-ref;
            description "P2P TE tunnel";
          }
        }
        case tunnel-p2mp {
          leaf p2mp-id {
            type te: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 8: TE generic YANG module

```

<CODE BEGINS> file "ietf-te-device@2018-02-15.yang"
module ietf-te-device {

  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;
}

/* Import TE generic types */
import ietf-te-types {
    prefix te-types;
}

import ietf-interfaces {
    prefix if;
}

import ietf-inet-types {
    prefix inet;
}

import ietf-routing-types {
    prefix "rt-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@cisco.com>

  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@ericsson.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
  "YANG data module for TE device configurations,
  state, RPC and notifications.";

revision "2018-02-15" {
  description "Latest update to TE device YANG module.";
  reference "TBA";
}

/**
 * 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";
    }
}
grouping lsp-device-timers {
    description "TE LSP timers configuration";
    uses lsp-device-timers_config;
}

/**
 * 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 te-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 MIRRORRED_UP_DOWN {
      description
        "MIRRORRED_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 te-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 te-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 te-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";
    }
}
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;
  }

  /* TE globals device augmentation */
  augment "/te:te/te:globals" {
    description
      "Global TE device specific configuration parameters";
    uses lsp-device-timers;
  }

  /* TE tunnels device configuration augmentation */
  augment "/te:te/te:tunnels/te:tunnel" {
    description
      "Tunnel device dependent augmentation";
    uses lsp-device-timers_config;
  }
  augment "/te:te/te:tunnels/te:tunnel/te:state" {
    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:state/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:state/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 9: TE MPLS specific types YANG module

```

<CODE BEGINS> file "ietf-te-mpls@2018-02-15.yang"
module ietf-te-mpls {

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

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

  /* Import TE base model */
  import ietf-te {
    prefix te;
  }

  /* Import TE MPLS types */
  import ietf-te-mpls-types {
    prefix "te-mpls-types";
  }

  /* Import TE generic types */
  import ietf-te-types {
    prefix te-types;
  }
}

```

```
/* Import routing types */
import ietf-routing-types {
  prefix "rt-types";
}

import ietf-mpls-static {
  prefix mpls-static;
}

import ietf-inet-types {
  prefix inet;
}

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@cisco.com>

  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@ericsson.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
  "YANG data module for MPLS TE configurations,
  state, RPC and notifications.";

revision "2018-02-15" {
  description "Latest update to MPLS TE YANG module.";
  reference "TBD";
}

/* MPLS TE tunnel properties*/

grouping tunnel-igp-shortcut_config {
  description "TE tunnel IGP shortcut configs";
  leaf shortcut-eligible {
    type boolean;
    default "true";
    description
      "Whether this LSP is considered to be eligible for us as a
      shortcut in the IGP. In the case that this leaf is set to
      true, the IGP SPF calculation uses the metric specified to
      determine whether traffic should be carried over this LSP";
  }
  leaf metric-type {
    type identityref {
      base te-types:LSP_METRIC_TYPE;
    }
    default te-types:LSP_METRIC_INHERITED;
    description
      "The type of metric specification that should be used to set
      the LSP(s) metric";
  }
  leaf metric {
    type int32;
    description
      "The value of the metric that should be specified. The value
      supplied in this leaf is used in conjunction with the metric
      type to determine the value of the metric used by the system.
      Where the metric-type is set to LSP_METRIC_ABSOLUTE - the
      value of this leaf is used directly; where it is set to
      LSP_METRIC_RELATIVE, the relevant (positive or negative)
      offset is used to formulate the metric; where metric-type
      is LSP_METRIC_INHERITED, the value of this leaf is not
      utilised";
  }
  leaf-list routing-afs {
```



```
    type inet:ip-version;
    description
      "Address families";
  }
}

grouping tunnel-igp-shortcuts {
  description
    "TE tunnel IGP shortcut grouping";
  container tunnel-igp-shortcut {
    description
      "Tunnel IGP shortcut properties";
    uses tunnel-igp-shortcut_config;
  }
}

grouping tunnel-forwarding-adjacency_configs {
  description "Tunnel forwarding adjacency grouping";
  leaf binding-label {
    type rt-types:mpls-label;
    description "MPLS tunnel binding label";
  }
  leaf load-share {
    type uint32 {
      range "1..4294967295";
    }
    description "ECMP tunnel forwarding
      load-share factor.";
  }
  leaf policy-class {
    type uint8 {
      range "1..7";
    }
    description
      "The class associated with this tunnel";
  }
}

grouping tunnel-forwarding-adjacency {
  description "Properties for using tunnel in forwarding.";
  container forwarding {
    description
      "Tunnel forwarding properties container";
    uses tunnel-forwarding-adjacency_configs;
  }
}

/** End of MPLS TE tunnel configuration/state */
```

```
grouping te-lsp-auto-bandwidth_config {
  description
    "Configuration parameters related to autobandwidth";

  leaf enabled {
    type boolean;
    default false;
    description
      "enables mpls auto-bandwidth on the
       lsp";
  }

  leaf min-bw {
    type te-mpls-types:bandwidth-kbps;
    description
      "set the minimum bandwidth in Kbps for an
       auto-bandwidth LSP";
  }

  leaf max-bw {
    type te-mpls-types:bandwidth-kbps;
    description
      "set the maximum bandwidth in Kbps for an
       auto-bandwidth LSP";
  }

  leaf adjust-interval {
    type uint32;
    description
      "time in seconds between adjustments to
       LSP bandwidth";
  }

  leaf adjust-threshold {
    type te-types:percentage;
    description
      "percentage difference between the LSP's
       specified bandwidth and its current bandwidth
       allocation -- if the difference is greater than the
       specified percentage, auto-bandwidth adjustment is
       triggered";
  }
}

grouping te-lsp-overflow_config {
  description
    "configuration for mpls lsp bandwidth
     overflow adjustment";
```

```
leaf enabled {
    type boolean;
    default false;
    description
        "enables mpls lsp bandwidth overflow
        adjustment on the lsp";
}

leaf overflow-threshold {
    type te-types:percentage;
    description
        "bandwidth percentage change to trigger
        an overflow event";
}

leaf trigger-event-count {
    type uint16;
    description
        "number of consecutive overflow sample
        events needed to trigger an overflow adjustment";
}
}

grouping te-lsp-underflow_config {
    description
        "configuration for mpls lsp bandwidth
        underflow adjustment";

    leaf enabled {
        type boolean;
        default false;
        description
            "enables bandwidth underflow
            adjustment on the lsp";
    }

    leaf underflow-threshold {
        type te-types:percentage;
        description
            "bandwidth percentage change to trigger
            and underflow event";
    }

    leaf trigger-event-count {
        type uint16;
        description
            "number of consecutive underflow sample
```

```
        events needed to trigger an underflow adjustment";
    }
}
grouping te-tunnel-bandwidth_config {
    description
        "Configuration parameters related to bandwidth for a tunnel";

    leaf specification-type {
        type te-mpls-types:te-bandwidth-type;
        default SPECIFIED;
        description
            "The method used for settign the bandwidth, either explicitly
            specified or configured";
    }

    leaf set-bandwidth {
        when "../specification-type = 'SPECIFIED'" {
            description
                "The bandwidth value when bandwidth is explicitly
                specified";
        }
        type te-mpls-types:bandwidth-kbps;
        description
            "set bandwidth explicitly, e.g., using
            offline calculation";
    }

    leaf class-type {
        type te-types:te-ds-class;
        description
            "The Class-Type of traffic transported by the LSP.";
        reference "RFC4124: section-4.3.1";
    }
}

grouping te-tunnel-bandwidth_state {
    description
        "Operational state parameters relating to bandwidth for a tunnel";

    leaf signaled-bandwidth {
        type te-mpls-types:bandwidth-kbps;
        description
            "The currently signaled bandwidth of the LSP. In the case where
            the bandwidth is specified explicitly, then this will match the
            value of the set-bandwidth leaf; in cases where the bandwidth is
            dynamically computed by the system, the current value of the
            bandwidth should be reflected.";
    }
}
```

```
grouping tunnel-bandwidth_top {
  description
    "Top level grouping for specifying bandwidth for a tunnel";

  container bandwidth-mpls {
    description
      "Bandwidth configuration for TE LSPs";

    uses te-tunnel-bandwidth_config;

    container state {
      config false;
      description
        "State parameters related to bandwidth
        configuration of TE tunnels";
      uses te-tunnel-bandwidth_state;
    }

    container auto-bandwidth {
      when "../specification-type = 'AUTO'" {
        description
          "Include this container for auto bandwidth
          specific configuration";
      }
      description
        "Parameters related to auto-bandwidth";

      uses te-lsp-auto-bandwidth_config;

      container overflow {
        description
          "configuration of MPLS overflow bandwidth
          adjustement for the LSP";

        uses te-lsp-overflow_config;
      }

      container underflow {
        description
          "configuration of MPLS underflow bandwidth
          adjustement for the LSP";

        uses te-lsp-underflow_config;
      }
    }
  }
}
```

```
grouping te-path-bandwidth_top {
  description
    "Top level grouping for specifying bandwidth for a TE path";

  container bandwidth {
    description
      "Bandwidth configuration for TE LSPs";

    uses te-tunnel-bandwidth_config;
    container state {
      config false;
      description
        "State parameters related to bandwidth
        configuration of TE tunnels";
      uses te-tunnel-bandwidth_state;
    }
  }
}

/**
 * MPLS TE augmentations
 */

/* MPLS TE tunnel augmentations */
augment "/te:te/te:tunnels/te:tunnel" {
  description "MPLS TE tunnel config augmentations";
  uses tunnel-igp-shortcuts;
  uses tunnel-forwarding-adjacency;
  uses tunnel-bandwidth_top;
}

/* MPLS TE LSPs augmentations */
augment "/te:te/te:tunnels/te:tunnel/" +
  "te:p2p-primary-paths/te:p2p-primary-path" {
  when "/te:te/te:tunnels/te:tunnel" +
    "/te:p2p-primary-paths/te:p2p-primary-path" +
    "/te:path-setup-protocol = 'te-types:path-setup-static'" {
    description
      "When the path is statically provisioned";
  }
  description "MPLS TE LSP augmentation";
  leaf static-lsp-name {
    type mpls-static:static-lsp-ref;
    description "Static LSP name";
  }
}
```

```

augment "/te:te/te:tunnels/te:tunnel/" +
    "te:p2p-primary-paths/te:p2p-primary-path/" +
    "te:state" {
    description "MPLS TE LSP augmentation";
    leaf static-lsp-name {
        type mpls-static:static-lsp-ref;
        description "Static LSP name";
    }
}
augment "/te:te/te:tunnels/te:tunnel/" +
    "te:p2p-secondary-paths/te:p2p-secondary-path" {
    when "/te:te/te:tunnels/te:tunnel" +
        "/te:p2p-secondary-paths/te:p2p-secondary-path/" +
        "te:path-setup-protocol = 'te-types:path-setup-static'" {
        description
            "When the path is statically provisioned";
    }
    description "MPLS TE LSP augmentation";
    leaf static-lsp-name {
        type mpls-static:static-lsp-ref;
        description "Static LSP name";
    }
}
augment "/te:te/te:tunnels/te:tunnel/" +
    "te:p2p-secondary-paths/te:p2p-secondary-path/" +
    "te:state" {
    description "MPLS TE LSP augmentation";
    leaf static-lsp-name {
        type mpls-static:static-lsp-ref;
        description "Static LSP name";
    }
}

augment "/te:te/te:globals/te:named-path-constraints/" +
    "te:named-path-constraint" {
    description "foo";
    uses te-path-bandwidth_top;
}
}
<CODE ENDS>

```

Figure 10: TE MPLS YANG module

```

<CODE BEGINS> file "ietf-te-mpls-types@2018-02-15.yang"
module ietf-te-mpls-types {

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

```

```
/* Replace with IANA when assigned */
prefix "te-mpls-types";

organization
  "IETF TEAS Working Group";

contact "Fill me";

description
  "This module contains a collection of generally
  useful TE specific YANG data type defintions.";

revision "2018-02-15" {
  description "Latest revision of TE MPLS types";
  reference "RFC3209";
}

identity backup-protection-type {
  description
    "Base identity for backup protection type";
}

identity backup-protection-link {
  base backup-protection-type;
  description
    "backup provides link protection only";
}

identity backup-protection-node-link {
  base backup-protection-type;
  description
    "backup offers node (preferred) or link protection";
}

identity bc-model-type {
  description
    "Base identity for Diffserv-TE bandwidth constraint
    model type";
}

identity bc-model-rdm {
  base bc-model-type;
  description
    "Russian Doll bandwidth constraint model type.";
}

identity bc-model-mam {
  base bc-model-type;
```



```
    description
      "Maximum Allocation bandwidth constraint
      model type.";
  }

  identity bc-model-mar {
    base bc-model-type;
    description
      "Maximum Allocation with Reservation
      bandwidth constraint model type.";
  }

  typedef bandwidth-kbps {
    type uint64;
    units "Kbps";
    description
      "Bandwidth values expressed in kilobits per second";
  }

  typedef bandwidth-mbps {
    type uint64;
    units "Mbps";
    description
      "Bandwidth values expressed in megabits per second";
  }

  typedef bandwidth-gbps {
    type uint64;
    units "Gbps";
    description
      "Bandwidth values expressed in gigabits per second";
  }

  typedef te-bandwidth-type {
    type enumeration {
      enum SPECIFIED {
        description
          "Bandwidth is explicitly specified";
      }
      enum AUTO {
        description
          "Bandwidth is automatically computed";
      }
    }
    description
      "enumerated type for specifying whether bandwidth is
      explicitly specified or automatically computed";
  }
```

```
typedef bfd-type {
  type enumeration {
    enum classical {
      description "BFD classical session type.";
    }
    enum seamless {
      description "BFD seamless session type.";
    }
  }
  default "classical";
  description
    "Type of BFD session";
}

typedef bfd-encap-mode-type {
  type enumeration {
    enum gal {
      description
        "BFD with GAL mode";
    }
    enum ip {
      description
        "BFD with IP mode";
    }
  }
  default ip;
  description
    "Possible BFD transport modes when running over TE
    LSPs.";
}
}
<CODE ENDS>
```

Figure 11: TE MPLS types YANG module

```
<CODE BEGINS> file "ietf-te-sr-mpls@2018-02-15.yang"
module ietf-te-sr-mpls {

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

  /* Replace with IANA when assigned */
  prefix "te-sr-mpls";

  /* Import TE generic types */
  import ietf-te {
    prefix te;
  }
}
```

```
/* Import TE generic types */
import ietf-te-types {
  prefix te-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@cisco.com>

  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@ericsson.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
  "YANG data module for MPLS TE configurations,
  state, RPC and notifications.";

revision "2018-02-15" {
```

```
    description "Latest update to MPLS TE YANG module.";
    reference "TBD";
}

identity sr-protection-type {
    description
        "The Adj-SID base protection types";
}

identity sr-protection-type-protected {
    base sr-protection-type;
    description
        "The Adj-SID is eligible if protected";
}

identity sr-protection-type-unprotected {
    base sr-protection-type;
    description
        "The Adj-SID is eligible if unprotected";
}

identity sr-protection-type-any {
    base sr-protection-type;
    description
        "The Adj-SID is eligible if protected or unprotected";
}

typedef te-sid-selection-mode {
    type enumeration {
        enum ADJ_SID_ONLY {
            description
                "The SR-TE tunnel should only use adjacency SIDs
                to build the SID stack to be pushed for the LSP";
        }
        enum MIXED_MODE {
            description
                "The SR-TE tunnel can use a mix of adjacency
                and prefix SIDs to build the SID stack to be pushed
                to the LSP";
        }
    }
    description "SID selection mode type";
}

/* MPLS SR-TE tunnel properties*/
grouping tunnel-sr-mpls-properties_config {
    description "MPLS TE SR tunnel properties";
    leaf path-signaling-type {
```

```
    type identityref {
      base te-types:path-signaling-type;
    }
    description "TE tunnel path signaling type";
  }
}

grouping te-sr-named-path-constraints_config {
  description
    "Configuration parameters relating to SR-TE LSPs";

  leaf sid-selection-mode {
    type te-sid-selection-mode;
    default MIXED_MODE;
    description
      "The restrictions placed on the SIDs to be selected by the
      calculation method for the explicit path when it is
      instantiated for a SR-TE LSP";
  }

  leaf sid-protection {
    type identityref {
      base sr-protection-type;
    }
    default sr-protection-type-any;
    description
      "When set to protected only SIDs that are
      protected are to be selected by the calculating method
      when the explicit path is instantiated by a SR-TE LSP.";
  }
}

grouping te-sr-named-path-constraints {
  description "Named TE SR path constraints grouping";
  uses te-sr-named-path-constraints_config;
}

/** End of MPLS SR-TE tunnel configuration/state */

/**
 * MPLS TE augmentations
 */
augment "/te:te/te:globals/te:named-path-constraints" +
  "/te:named-path-constraint" {
  description
    "Augmentations for MPLS SR-TE config named constraints";
  uses te-sr-named-path-constraints;
}
```

```
/* MPLS TE tunnel augmentations */  
  
/* MPLS TE LSPs augmentations */  
}  
<CODE ENDS>
```

Figure 12: SR TE MPLS YANG module

5. IANA Considerations

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

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

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

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

URI: urn:ietf:params:xml:ns:yang:ietf-te-sr-mpls XML: N/A, the requested URI is an XML namespace.

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

URI: urn:ietf:params:xml:ns:yang:ietf-te-mpls-types XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-te namespace: urn:ietf:params:xml:ns:yang:ietf-te prefix: ietf-te reference: RFC3209

name: ietf-te-device namespace: urn:ietf:params:xml:ns:yang:ietf-te prefix: ietf-te-device reference: RFC3209

name: ietf-te-mpls namespace: urn:ietf:params:xml:ns:yang:ietf-te-mpls prefix: ietf-te-mpls reference: RFC3209

name: ietf-te-sr-mpls namespace: urn:ietf:params:xml:ns:yang:ietf-te-sr-mpls prefix: ietf-te-sr-mpls

name: ietf-te-types namespace: urn:ietf:params:xml:ns:yang:ietf-te-
types prefix: ietf-te-types reference: RFC3209

name: ietf-te-mpls-types namespace: urn:ietf:params:xml:ns:yang:ietf-
te-mpls-types prefix: ietf-te-mpls-types reference: RFC3209

6. Security Considerations

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

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. Normative References

- [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-09 (work in progress), May 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>>.
- [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>>.
- [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>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [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>>.
- [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>>.

Authors' Addresses

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

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

TEAS Working Group
Internet-Draft
Intended status: Informational
Expires: December 26, 2018

D. King (Ed.)
Old Dog Consulting
Y. Lee (Ed.)
Huawei

June 26, 2018

Applicability of Abstraction and Control
of Traffic Engineered Networks (ACTN) to Network Slicing
draft-king-teas-applicability-actn-slicing-03

Abstract

Network abstraction is a technique that can be applied to a network domain to select network resources by policy to obtain a view of potential connectivity

Network slicing is an approach to network operations that builds on the concept of network abstraction to provide programmability, flexibility, and modularity. It may use techniques such as Software Defined Networking (SDN) and Network Function Virtualization (NFV) to create multiple logical (virtual) networks, each tailored for a set of services that are sharing the same set of requirements, on top of a common network.

These logical networks are referred to as transport network slices. A transport network slice does not necessarily represent dedicated resources in the network, but does constitute a commitment by the network provider to provide a specific level of service.

The Abstraction and Control of Traffic Engineered Networks (ACTN) defines an SDN-based architecture that relies on the concepts of network and service abstraction to detach network and service control from the underlying data plane.

This document outlines the applicability of ACTN to transport network slicing in an IETF technology network. It also identifies the features of network slicing not currently within the scope of ACTN, and indicates where ACTN might be extended.

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 December 26 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction.....	3
1.1. Terminology.....	4
2. Requirements for Network Slicing.....	4
2.1. Resource Slicing.....	4
2.2. Network and Function Virtualization.....	5
2.3. Resource Isolation.....	5
2.4. Control and Orchestration.....	6
3. Abstraction and Control of Traffic Engineered (TE) Networks (ACTN).....	6
3.1. ACTN Virtual Network as a "Network Slice".....	8
3.2. Examples of ACTN Delivering Types of Network Slices.....	8
3.2.1. ACTN Used for Virtual Private Line Model.....	9
3.2.2. ACTN Used for VPN Delivery Model.....	10
3.2.3. ACTN Used to Deliver a Virtual Customer Network.....	10
3.3. Network Slice Service Mapping from TE to ACTN VN Models....	11
3.4. ACTN VN KPI Telemetry Models.....	12
4. IANA Considerations.....	12
5. Security Considerations.....	12
6. Acknowledgements.....	12
7. References.....	13
8. Contributors.....	15
Authors' Addresses.....	15

1. Introduction

The principles of network resource separation are not new. For years, separated overlay and logical (virtual) networking have existed, allowing multiple connectivity services to be deployed over a single physical network comprised of single or multiple layers. However, several key differences exist that differentiate overlay and virtual networking from network slicing.

A transport network slice construct provides an end-to-end logical network, often with compute functions and utilising shared underlying (physical or virtual) network resources. This logical network is separated from other, often concurrent, logical networks each with independent control and management, and each of which can be created or modified on demand.

At one end of the spectrum, a virtual private wire or a virtual private network (VPN) may be used to build a network slice. In these cases, the network slices do not require the service provider to isolate network resources for the provision of the service - the service is "virtual".

At the other end of the spectrum there may be a detailed description of a complex service that will meet the needs of a set of applications with connectivity and service function requirements that may include compute resource, storage capability, and access to content. Such a service may be requested dynamically (that is, instantiated when an application needs it, and released when the application no longer needs it), and modified as the needs of the application change.

Each example represents a self-contained network that must be flexible enough to simultaneously accommodate diverse business-driven use cases from multiple players on a common network infrastructure.

This document outlines the application of the ACTN architecture [actn-framework] and enabling technologies to provide transport network slicing in an IETF technology network. It describes how the ACTN functional components can be used to support model-driven partitioning of variable-sized bandwidth to facilitate network sharing and virtualization. Furthermore, the use of model-based interfaces to dynamically request the instantiation of virtual networks could be extended to encompass requesting and instantiation of specific service functions (which may be both physical and/or virtual), and to partition network resources such as compute resource, storage capability, and access to content.

In an IETF context, there are works in progress that have some bearing with network slicing such as Enhanced VPN (VPN+) and DetNet. Both works are an independent work in their own scope while

This document highlights how the ACTN approach might be extended to address these other requirements of network slicing where TE is required.

1.1. Terminology

Resource: Any features that can be delivered, including connectivity, compute, storage, and content delivery.

Service Functions (SFs): Components that provide specific function within a network. SFs are often combined in a specific sequence, service function chain, to deliver services.

Infrastructure Resources: The hardware and necessary software for hosting and connecting SFs. These resources may include computing hardware, storage capacity, network resources (e.g. links and switching/routing devices enabling network connectivity), and physical assets for radio access.

Service Provider: A server network or collection of server networks.

Consumer: Any application, client network, or customer of a network provider.

Service Level Agreement (SLA): An agreement between a consumer and network provider that describes the quality with which features and functions are to be delivered. It may include measures of bandwidth, latency, and jitter; the types of service (such as the network service functions or billing) to be executed; the location, nature, and quantities of services (such as the amount and location of compute resources and the accelerators require).

Network Slice: An agreement between a consumer and a service provider to deliver network resources according to a specific service level agreement. A slice could span multiple technology (e.g., radio, transport and cloud) and administrative domains.

IETF Technology: A TE network slice or transport network slice.

2. Requirements for Network Slicing

The concept of network slicing is considered a key capability for future networks and, to serve customers with a wide variety of different service needs, in term of latency, reliability, capacity, and service function specific capabilities.

This section outlines the key capabilities required, and further discussed in [ngmn-network-slicing], [network-slice-5g], [3gpp.28.801] and [onf-tr526], to realise network slicing in an IETF technology network.

2.1. Resource Slicing

For network slicing, it is important to consider both infrastructure resources and service functions. This allows a flexible approach to deliver a range of services both by partitioning (slicing) the available network resources to present them for use by a consumer, but also by providing instances of SFs at the right locations and in the correct chaining logic, with access to the necessary hardware, including specific compute and storage resources.

Mapping of resources to slices may 1-to-1, or resources may be shared among multiple slices.

2.2. Network and Function Virtualization

Virtualization is the abstraction of resources where the abstraction is made available for use by an operations entity, for example, by the Network Management Station (NMS) of a consumer network. The resources to be virtualized can be physical or already virtualized, supporting a recursive pattern with different abstraction layers. Therefore, Virtualization is critical for network slicing as it enables effective resource sharing between network slices.

Just as server virtualization makes virtual machines (VMs) independent of the underlying physical hardware, network Virtualization enables the creation of multiple isolated virtual networks that are completely decoupled from the underlying physical network, and can safely run on top of it.

2.3. Resource Isolation

Isolation of data and traffic is a major requirement that must be satisfied for certain applications to operate in concurrent network slices on a common shared underlying infrastructure. Therefore, isolation must be understood in terms of:

- o Performance: Each slice is defined to meet specific service requirements, usually expressed in the form of Key Performance Indicators (KPIs). Performance isolation requires that service delivery on one network slice is not adversely impacted by congestion and performance levels of other slices;
- o Security: Attacks or faults occurring in one slice must not have an impact on other slices, or customer flows are not only isolated on network edge, but multiple customer traffic is not mixed across the core of the network. Moreover, each slice must have independent security functions that prevent unauthorised entities to have read or write access to slice-specific configuration, management, accounting information, and able to record any of these attempts, whether authorised or not;

- o Management: Each slice must be independently viewed, utilised and managed as a separate network.

2.4. Control and Orchestration

Orchestration is the overriding control method for network slicing. We may define orchestration as combining and coordinating multiple control methods to provide an operational mechanism that can deliver services and control underlying resources. In a network slicing environment, an orchestrator is needed to coordinate disparate processes and resources for creating, managing, and deploying the end-to-end service. Two scenarios are outlined below where orchestration would be required:

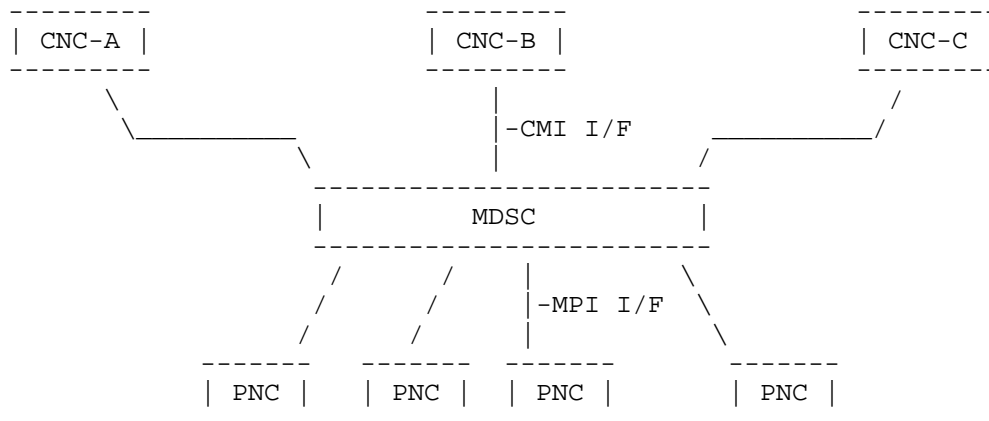
1. Multi-domain Orchestration: Managing connectivity setup of the transport service, across multiple administrative domains;
2. End-to-end Orchestration: Combining resources for an "end-to-end service (e.g., transport connectivity with firewalling and guaranteed bandwidth and minimum delay for premium radio users (spanning multiple domains)).

In addition, 3GPP has also developed Release 14 "Study on management and orchestration of network slicing for next generation network" [3gpp.28.801], which defines an information model where the network slice as well as physical and virtualized network functions belong to the network operator domain, while the virtualized resources belong to another domain operated by a Virtualization infrastructure service provider.

3. Abstraction and Control of Traffic Engineered (TE) Networks (ACTN)

The framework for ACTN [actn-framework] includes a reference architecture that has been adapted for Figure 1 in this document, it describes the functional entities and methods for the coordination of resources across multiple domains, to provide end-to-end services, components include:

- o Customer Network Controller (CNC);
- o Multi-domain Service Coordinator (MDSC);
- o Provisioning Network Controller (PNC).



CMI - (CNC-MDSC Interface)
 MPI - (MDSC-PNC Interface)

Figure 1: ACTN Hierarchy

ACTN facilitates end-to-end connections and provides them to the user. The ACTN framework highlights how:

- o Abstraction of the underlying network resources are provided to higher-layer applications and customers;
- o Virtualization of underlying resources, whose selection criterion is the allocation of those resources for the customer, application, or service;
- o Creation of a virtualized environment allowing operators to view and control multi-domain networks as a single virtualized network;
- o The presentation to customers of networks as a virtual network via open and programmable interfaces.

The ACTN managed infrastructure are traffic engineered network resources, which may include:

- o Statistical packet bandwidth;
- o Physical forwarding plane sources, such as: wavelengths and time slots;
- o Forwarding and cross connect capabilities.

The ACTN type of network virtualization provides customers and applications (tenants) to utilise and independently control

allocated virtual network resources as if resources as if they were physically their own resource. The ACTN network is "sliced", with tenants being given a different partial and abstracted topology view of the physical underlying network. The capabilities that ACTN provides to enable slicing are outlined in Section 2 (Requirements for Network Slicing).

3.1. ACTN Virtual Network as a "Network Slice"

To support multiple clients each with its own view of and control of the server network, a network operator needs to partition (or "slice") the network resources. The resulting slices can be assigned to each client for guaranteed usage which is a step further than shared use of common network resources. See [actn-vn] for detailed ACTN VN and VNS.

An ACTN Virtual Network (VN) is a client view that may be considered a "network slice" of the ACTN managed infrastructure, and is presented by the ACTN provider as a set of abstracted resources.

Depending on the agreement between client and provider various VN operations and VN views are possible.

- o Network Slice Creation: A VN could be pre-configured and created via static or dynamic request and negotiation between customer and provider. It must meet the specified SLA attributes which satisfy the customer's objectives.
- o Network Slice Operations: The network slice may be further modified and deleted based on customer request to request changes in the network resources reserved for the customer, and used to construct the network slice. The customer can further act upon the network slice to manage traffic flow across the network slice.
- o Network Slice View: The VN topology from a customer point of view. These may be a variety of tunnels, or an entire VN topology. Such connections may comprise of customer end points, access links, intra domain paths and inter-domain links.

Primitives (capabilities and messages) have been provided to support the different ACTN network control functions that will enable network slicing. These include: topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options. [actn-info]

3.2. Examples of ACTN Delivering Types of Network Slices

In examples below the ACTN framework is used to provide

control, management and orchestration for the network slice life-cycle, the connectivity . These dynamic and highly flexible, end-to-end and dedicated network slices utilising common physical infrastructure, and according to vertical-specific requirements.

The rest of this section provides three examples of using ACTN to achieve different scenarios of ACTN for network slicing. All three scenarios can be scaled up in capacity or be subject to topology changes as well as changes from customer requirements perspective.

3.2.1. ACTN Used for Virtual Private Line Model

ACTN Provides virtual connections between multiple customer locations, requested via Virtual Private Line (VPL) requester (CNC-A). Benefits of this model include:

- o Automated: the service set-up and operation is network provider managed;
- o Virtual: the private line is seamlessly extended from customers Site A (vCE1 to vCE2) and Site B (vCE2 to vCE3) across the ACTN-managed WAN to Site C;
- o Agile: on-demand where the customer needs connectivity and fully adjustable bandwidth.

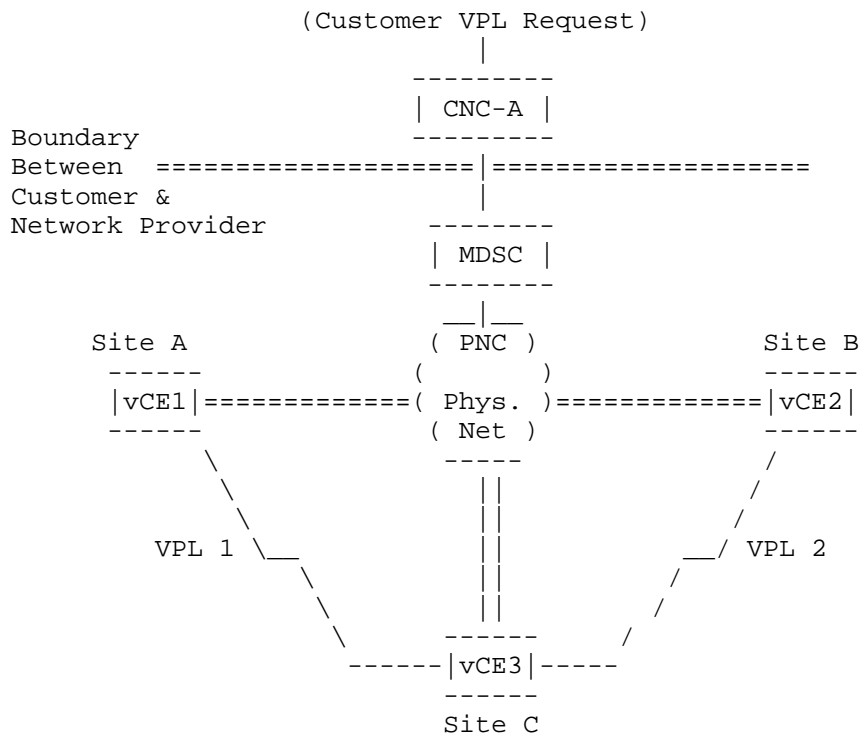


Figure 2: Virtual Private Line Model

3.2.2. ACTN Used for VPN Delivery Model

ACTN Provides VPN connections between multiple sites, requested via a VPN requestor (CNC-A), which is managed by the customer themselves. The CNC will then interact with the network providers MDSC. Benefits of this model include:

- o Provides edge-to-edge VPN multi-access connection;
- o Mostly network provider managed, with some flexibility delegated to the customer managed CNC.

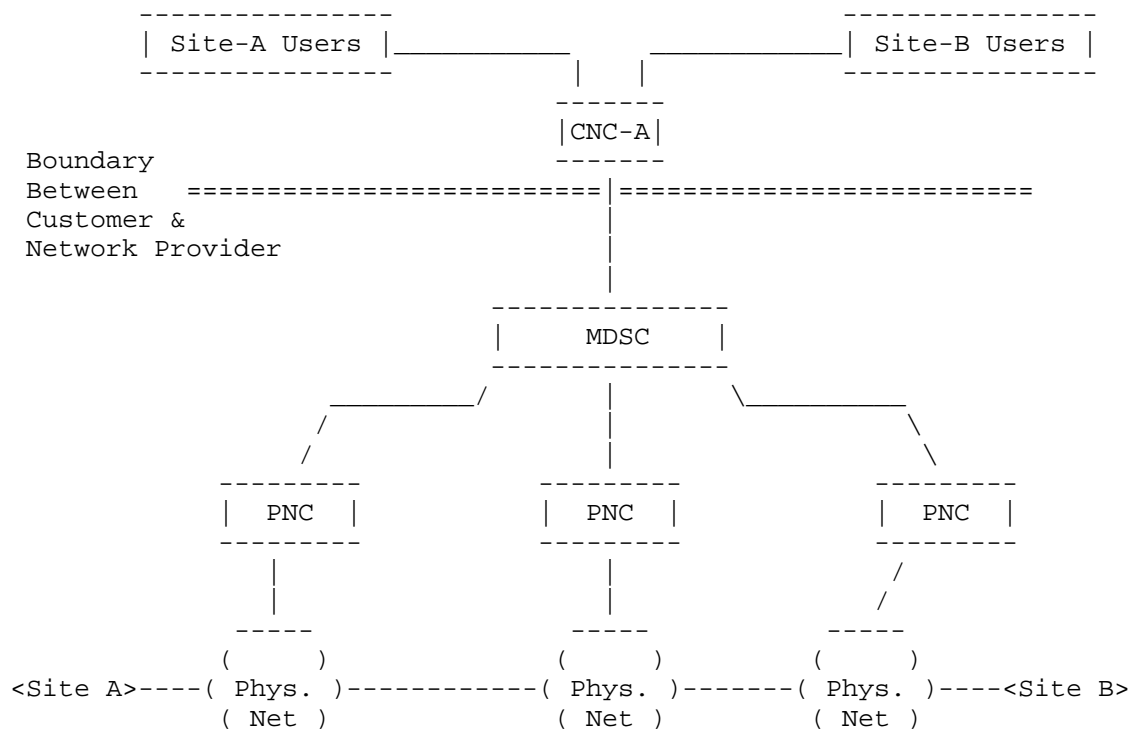


Figure 3: VPN Model

3.2.3. ACTN Used to Deliver a Virtual Customer Network

In this example ACTN provides a virtual network resource to the

customer. This resource is customer managed. Empowering the tenant to control allocated slice (recursively). Benefits of this model include:

- o The MDSC provides the topology as part of the customer view so that the customer can control their network slice to fit their needs;
- o Resource isolation, each customer network slice is fixed and will not be affected by changes to other customer network slices;
- o Applications can interact with their assigned network slice directly, the customer may implement their own network control method and traffic prioritization, manage their own addressing scheme, and further slice their assigned network resource;
- o The network slice may also include specific capability nodes, delivered as Physical Network Functions (PNFs) or Virtual Network Functions (VNFs).

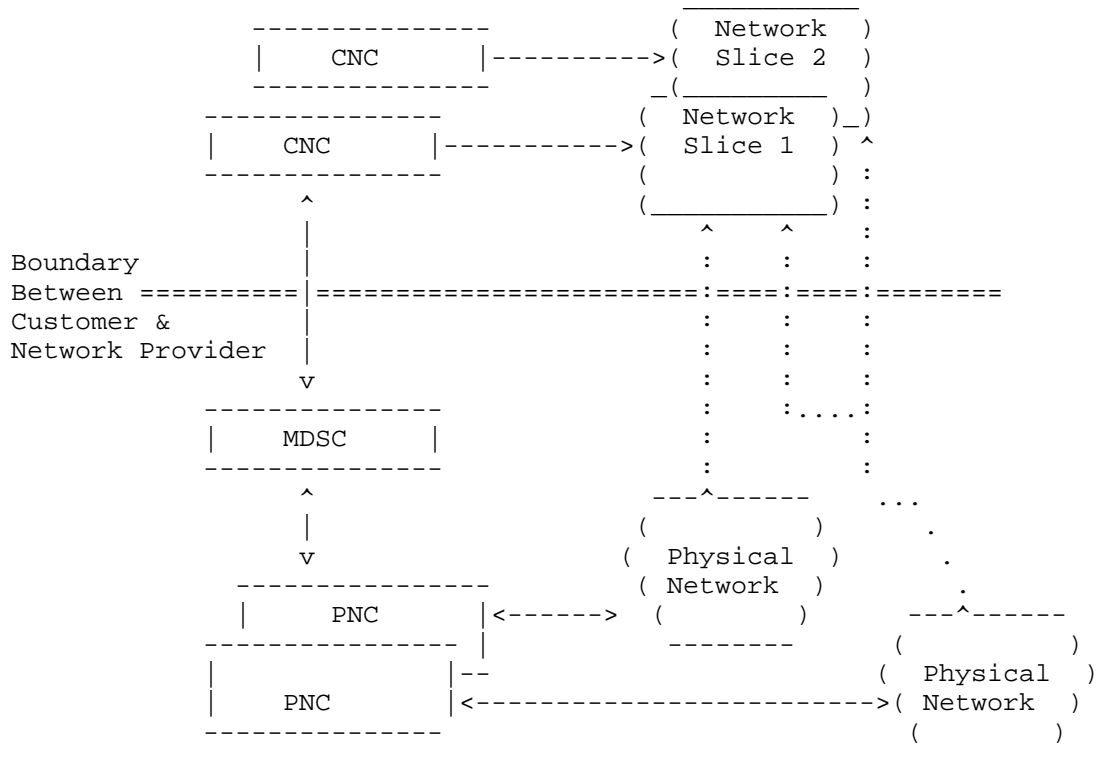


Figure 4: Network Slicing

3.3. Network Slice Service Mapping from TE to ACTN VN Models

The role of TE-service mapping model [te-service-mapping] is to create a binding relationship across a Layer-3 Service Model [l3sm], Layer-2 Service Model and TE Tunnel model, via a generic ACTN Virtual Network (VN) model [actn-vn].

The ACTN VN YANG model is a generic virtual network service model that allows customers (internal or external) to create a VN that meets the customer's service objective with various constraints.

The TE-service mapping model is needed to bind L3VPN specific service model with TE-specific parameters. This binding will facilitate a seamless service operation with underlay-TE network visibility. The TE-service model developed in this document can also be extended to support other services including L2SM, and L1CSM network service models.

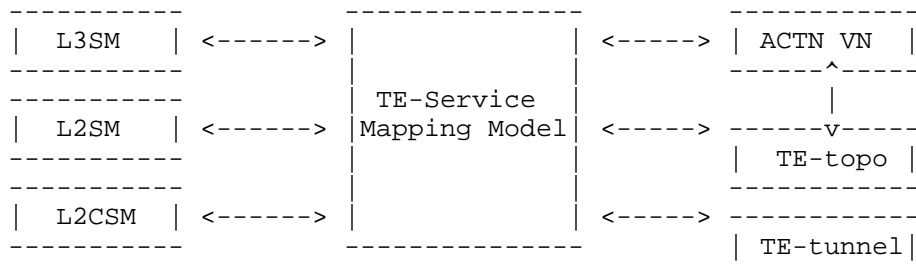


Figure 5: TE-Service Mapping ([te-service-mapping])

Editors note - We plan to provide a list of models available and their relationships/dependencies. We will also provide a vertical hierarchy of how these models may be used between functional components in ACTN.

3.4. ACTN VN KPI telemetry Models

The role of ACTN VN KPI telemetry model [actn-pm-telemetry] is to provide YANG models so that customer can define key performance monitoring data relevant for its VN/network slicing via the YANG subscription model.

Key characteristics of [actn-pm-telemetry] include:

- o an ability to provide scalable VN-level telemetry aggregation based on customer-subscription model for key performance parameters defined by the customer;
- o an ability to facilitate proactive re-optimization and reconfiguration of VNs/Network Slices based on network autonomic traffic engineering scaling configuration mechanism.

5. IANA Considerations

This document makes no requests for action by IANA.

6. Security Considerations

Network slicing involves the control of network resources in order to meet the service requirements of consumers. In some deployment models, the consumer is able to directly request modification in the behaviour of resources owned and operated by a service provider. Such changes could significantly affect the service provider's ability to provide services to other consumers. Furthermore, the resources allocated for or consumed by a consumer will normally be billable by the service provider.

Therefore, it is crucial that the mechanisms used in any network slicing system allow for authentication of requests, security of those requests, and tracking of resource allocations.

It should also be noted that while the partitioning or slicing of resources is virtual, the consumers expect and require that there is no risk of leakage of data from one slice to another, no transfer of knowledge of the structure or even existence of other slices, and that changes to one slice (under the control of one consumer) should not have detrimental effects on the operation of other slices (whether under control of different or the same consumers) beyond the limits allowed within the SLA. Thus, slices are assumed to be private and to provide the appearance of genuine physical connectivity.

ACTN operates using the [netconf] or [restconf] protocols and assumes the security characteristics of those protocols. Deployment models for ACTN should fully explore the authentication and other security aspects before networks start to carry live traffic.

7. Acknowledgements

Thanks to Qin Wu, Andy Jones, Ramon Casellas, and Gert Grammel for their insight and useful discussions about network slicing.

8. References

8.1. Normative References

8.2. Informative References

- [ngmn-network-slicing]
NGMN, "Description of Network Slicing Concept", 1 2016,
<[https://www.ngmn.org/uploads/
media/160113_Network_Slicing_v1_0.pdf](https://www.ngmn.org/uploads/media/160113_Network_Slicing_v1_0.pdf)>.
- [3gpp.28.801]
3GPP, "Study on management and orchestration of network
slicing for next generation network", 3GPP TR 28.801
0.4.0, 1 2017,
<<http://www.3gpp.org/ftp/Specs/html-info/28801.htm>>.
- [network-slice-5g]
"Network Slicing for 5G with SDN/NFV: Concepts,
Architectures and Challenges", Jose Ordonez-Lucena,
Pablo Ameigeiras, Diego Lopez, Juan J. Ramos-Munoz,
Javier Lorca, Jesus Folgueira, IEEE Communications
Magazine 55, March 2017
- [onf-tr526]
ONF TR-526, "Applying SDN Architecture to 5G Slicing",
April 2016.
- [actn-framework]
Ceccarelli, D. and Y. Lee, "Framework for Abstraction and
Control of Traffic Engineered Networks", draft-ietf-teas-
actn-framework, work in progress, February 2017.
- [te-service-mapping]
Y. Lee, D. Dhody, and D. Ceccarelli, "Traffic Engineering
and Service Mapping Yang Model",
draft-lee-teas-te-service-mapping-yang, work in progress.
- [actn-vn] Y. Lee (Editor), "A Yang Data Model for ACTN VN
Operation", draft-lee-teas-actn-vn-yang, work in progress.
- [actn-info] Y. Lee, S. Belotti (Editors), "Information Model for
Abstraction and Control of TE Networks (ACTN)", draft-ietf-
teas-actn-info-model, work in progress.
- [actn-pm-elemetry] Y. Lee, et al, "YANG models for ACTN TE
Performance Monitoring Telemetry and Network Autonomics",
draft-lee-teas-actn-pm-telemetry-autonomics, work in
progress.
- [l3sm] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data
Model for L3VPN Service Delivery", RFC 8049, February 2017

[netconf] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
and A. Bierman, Ed., "Network Configuration Protocol
(NETCONF)", RFC 6241.

[restconf] A. Bierman, M. Bjorklund, and K. Watsen, "RESTCONF
Protocol", draft-ietf-netconf-restconf, work in progress.

[sf-topology] I. Bryskin, et al, "Use Cases for SF Aware Topology
Models", draft-ietf-teas-use-cases-sf-aware-topo-model, work
in progress.

[vpn+] S. Bryant and J. Dong, "Enhanced Virtual Private Networks
(VPN+)", draft-bryant-rtgwg-enhanced-vpn, work in progress.

9. Contributors

The following people contributed text to this document.

Adrian Farrel
Email: afarrel@juniper.net

Mohamed Boucadair
Email: mohamed.boucadair@orange.com

Sergio Belotti
Email: sergio.belotti@nokia.com

Daniele Ceccarelli
Email: daniele.ceccarelli@ericsson.com

Haomian Zheng
Email: zhenghaomian@huawei.com

Authors' Addresses

Daniel King
Email: daniel@olddog.co.uk

Young Lee
Email: leeyoung@huawei.com

TEAS Working Group
Internet Draft
Intended Status: Standard Track
Expires: January 2, 2019

Y. Lee (Editor)
Dhruv Dhody
Satish Karunanithi
Huawei
Ricard Vilalta
CTTC
Daniel King
Lancaster University
Daniele Ceccarelli
Ericsson

July 2, 2018

YANG models for ACTN TE Performance Monitoring Telemetry and Network
Autonomics

draft-lee-teas-actn-pm-telemetry-autonomics-07

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/lid-abstracts.txt>

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

This Internet-Draft will expire on January 2, 2019.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents
(<http://trustee.ietf.org/license-info>) in effect on the date of
Lee, et al. Expires January 2019 [Page 1]

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

Abstraction and Control of TE Networks (ACTN) refers to the set of virtual network operations needed to operate, control and manage large-scale multi-domain, multi-layer and multi-vendor TE networks, so as to facilitate network programmability, automation, efficient resource sharing.

This document provides YANG data models that describe Key Performance Indicator (KPI) telemetry and network autonomics for TE-tunnels and ACTN VNs.

Table of Contents

1. Introduction.....	3
1.1. Terminology.....	3
1.2. Tree Structure - Legend.....	3
2. Use-Cases.....	4
3. Design of the Data Models.....	5
3.1. TE KPI Telemetry Model.....	6
3.2. ACTN TE KPI Telemetry Model.....	6
4. Notification.....	8
4.1. YANG Push Subscription Examples.....	8
5. YANG Data Tree.....	9
6. Yang Data Model.....	11
6.1. ietf-te-kpi-telemetry model.....	11
6.2. ietf-actn-te-kpi-telemetry model.....	19
7. Security Considerations.....	22
8. IANA Considerations.....	22
9. Acknowledgements.....	22
10. References.....	22
10.1. Informative References.....	22
10.2. Normative References.....	23
11. Contributors.....	24
Authors' Addresses.....	24

1. Introduction

Abstraction and Control of TE Networks (ACTN) describes a method for operating a Traffic Engineered (TE) network (such as an MPLS-TE network or a layer 1/0 transport network) to provide connectivity and virtual network services for customers of the TE network [ACTN-Frame]. The services provided can be optimized to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers. Data models are a representation of objects that can be configured or monitored within a system. Within the IETF, YANG [RFC6020] is the language of choice for documenting data models, and YANG models have been produced to allow configuration or modeling of a variety of network devices, protocol instances, and network services. YANG data models have been classified in [Netmod-Yang-Model-Classification] and [Service-YANG].

[ACTN-VN] describes how customers or end to end orchestrators can request and/or instantiate a generic virtual network service. [ACTN-Applicability] describes a connection between IETF YANG model classifications to ACTN interfaces. In particular, it describes the customer service model can be mapped into the CMI (CNC-MDSC Interface) of the ACTN architecture.

The YANG model on the ACTN CMI is known as customer service model in [Service-YANG]. [PCEP-Service-Aware] describes key network performance data to be considered for end-to-end path computation in TE networks. Key performance indicator is a term that describes critical performance data that may affect VN/TE service.

1.1. Terminology

1.2. Tree Structure - Legend

A simplified graphical representation of the data model is used in Section 5 of this document. The meaning of the symbols in these diagrams is defined in [RFC8342].

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
rt	ietf-routing-types	[Routing-Types]
te	ietf-te	[TE-tunnel]
te-types	ietf-te-types	[TE-Types]
te-kpi	ietf-te-kpi-telemetry	[This I-D]
vn	ietf-actn-vn	[ACTN-VN]
actn-tel	ietf-actn-te-kpi-telemetry	[This I-D]

Table 1: Prefixes and corresponding YANG modules

2. Use-Cases

[ACTN-PERF] describes use-cases relevant to this draft. It introduces the dynamic creation, modification and optimization of services based on the performance monitoring in the Abstraction and Control of Transport Networks (ACTN) architecture. Figure 1 shows a high-level workflows for dynamic service control based on traffic monitoring.

Some of the key points from [ACTN-PERF] are as follows:

- . Network traffic monitoring is important to facilitate automatic discovery of the imbalance of network traffic, and initiate the network optimization, thus helping the network operator or the virtual network service provider to use the network more efficiently and save CAPEX/OPEX.
- . Customer services have various SLA requirements, such as service availability, latency, latency jitter, packet loss rate, BER, etc. The transport network can satisfy service availability and BER requirements by providing different protection and restoration mechanisms. However, for other performance parameters, there are no such mechanisms. In order to provide high quality services according to customer SLA, one possible solution is to measure the service SLA related performance parameters, and dynamically provision and optimize services based on the performance monitoring results.
- . Performance monitoring in a large scale network could generate a huge amount of performance information. Therefore, the appropriate way to deliver the information in CMI and MPI interfaces should be carefully considered.

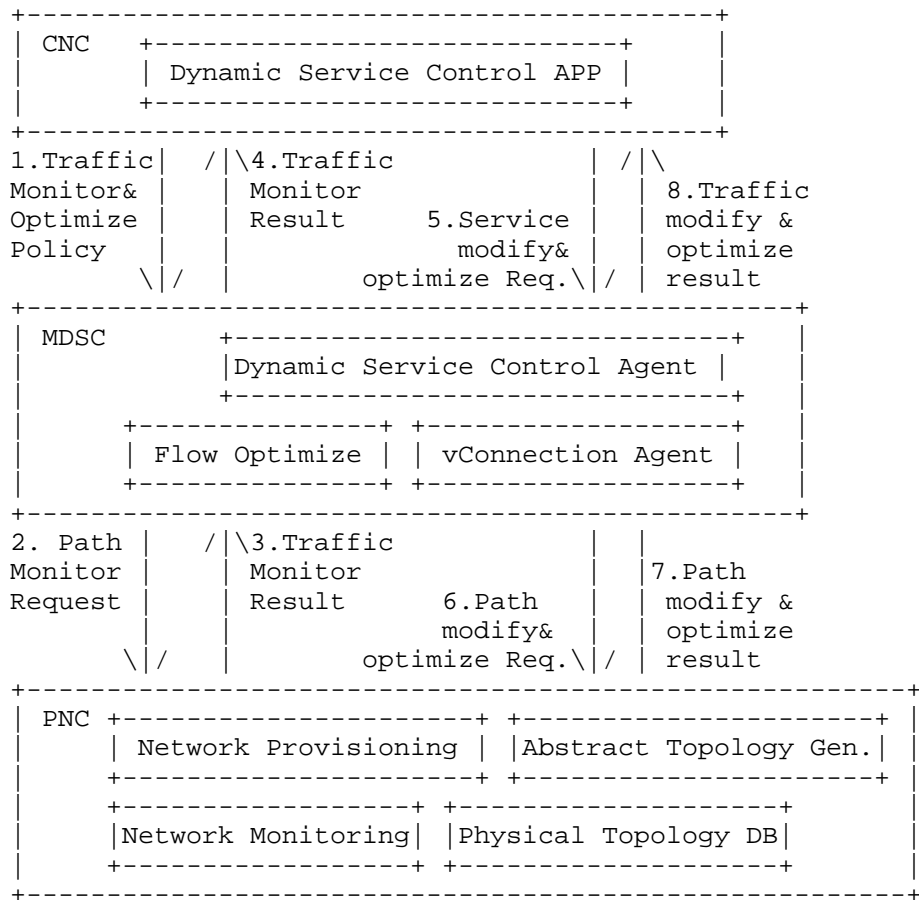


Figure 1 Workflows for dynamic service control based on traffic monitoring

3. Design of the Data Models

The YANG models developed in this document describe two models:

- (i) TE KPI Telemetry Model which provides the TE-Tunnel level of performance monitoring mechanism (See Section 4 for details)
- (ii) ACTN TE KPI Telemetry Model which provides the VN level of the aggregated performance monitoring mechanism (See Section 5 for details)

The models include -

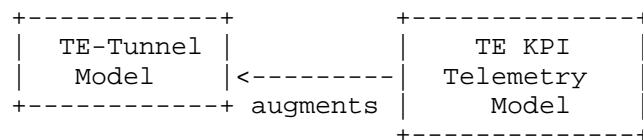
- (i) Performance Telemetry details as measured during the last interval, ex delay.
- (ii) Scaling Intent based on with TE/VN could be scaled in/out.

[Editor's Note - Need to decide if scaling and telemetry can be in the same model as per the current draft.]

3.1. TE KPI Telemetry Model

This module describes performance telemetry for TE-tunnel model. The telemetry data is augmented to tunnel state. This module also allows autonomic traffic engineering scaling intent configuration mechanism on the TE-tunnel level. Various conditions can be set for auto-scaling based on the telemetry data.

The TE KPI Telemetry Model augments the TE-Tunnel Model to enhance TE performance monitoring capability. This monitoring capability will facilitate proactive re-optimization and reconfiguration of TEs based on the performance monitoring data collected via the TE KPI Telemetry YANG model.

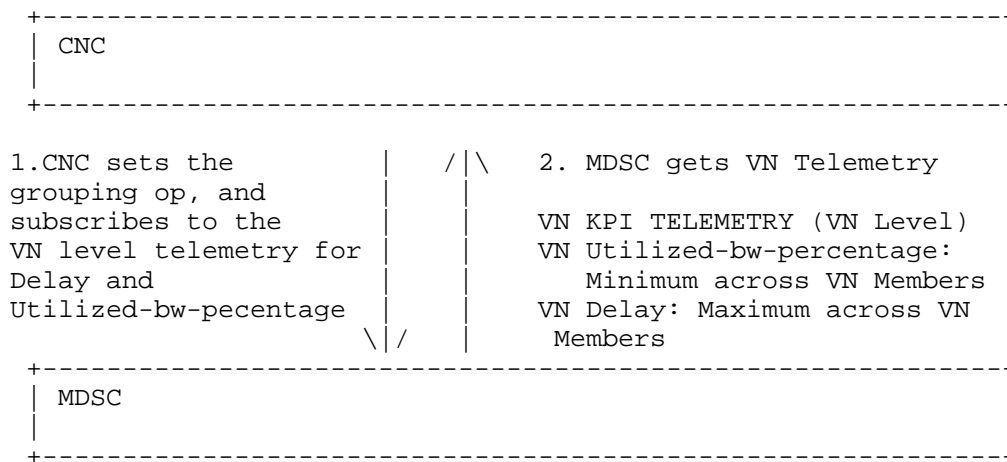


3.2. ACTN TE KPI Telemetry Model

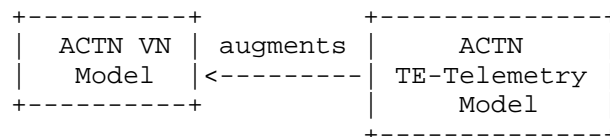
This module describes performance telemetry for ACTN VN model. The telemetry data is augmented both at the VN Level as well as individual VN member level. This module also allows autonomic traffic engineering scaling intent configuration mechanism on the VN

level. Scale in/out criteria might be used for network autonomics in order the controller to react to a certain set of variations in monitored parameters.

Moreover, this module also provides mechanism to define aggregated telemetry parameters as a grouping of underlying VN level telemetry parameters. Grouping operation (such as maximum, mean) could be set at the time of configuration. For example, if maximum grouping operation is used for delay at the VN level, the VN telemetry data is reported as the maximum {delay_vn_member_1, delay_vn_member_2,... delay_vn_member_N}. Thus, this telemetry abstraction mechanism allows the grouping of a certain common set of telemetry values under a grouping operation. This can be done at the VN-member level to suggest how the E2E telemetry be inferred from the per domain tunnel created and monitored by PNCs. One proposed example is the following:



The ACTN VN TE-Telemetry Model augments the basic ACTN VN model to enhance VN monitoring capability. This monitoring capability will facilitate proactive re-optimization and reconfiguration of VNs based on the performance monitoring data collected via the ACTN VN Telemetry YANG model.



4. Notification

This model does not define specific notifications. To enable notifications, the mechanism defined in [I-D.ietf-netconf-yang-push] and [I-D.ietf-netconf-rfc5277bis] can be used. 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.

4.1. YANG Push Subscription Examples

Below example shows the way for a client to subscribe for the telemetry information for a particular tunnel (Tunnell). The telemetry parameter that the client is interested in is the utilized bandwidth percentage.

```
<netconf:rpc netconf:message-id="101"
  xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push:1.0">
    <filter netconf:type="subtree">
      <te xmlns="urn:ietf:params:xml:ns:yang:ietf-te">
        <tunnels>
          <tunnel>
            <name>Tunnell</name>
            <identifier/>
            <state>
              <te-telemetry
xmlns="urn:ietf:params:xml:ns:yang:ietf-te-kpi-telemetry">
                <utilized-
percentage/>

              </te-telemetry>
            </state>
          </tunnel>
```

```

        </tunnels>
      </te>
    </filter>
    <period>500</period>
    <encoding>encode-xml</encoding>
  </establish-subscription>
</netconf:rpc>

```

This example shows the way for a client to subscribe for the telemetry information for all VNs. The telemetry parameter that the client is interested in is one-way delay and utilized bandwidth percentage.

```

<netconf:rpc netconf:message-id="101"
  xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push:1.0">
    <filter netconf:type="subtree">
      <actn-state xmlns="urn:ietf:params:xml:ns:yang:ietf-actn-
vn">
        <vn>
          <vn-list>
            <vn-id/>
            <vn-name/>
            <vn-
telemetry xmlns="urn:ietf:params:xml:ns:yang:ietf-actn-te-kpi-
telemetry">
              <one-way-delay/>
              <utilized-
percentage/>
            </vn-telemetry >
          </vn-list>
        </vn>
      </actn-state>
    </filter>
    <period>500</period>
  </establish-subscription>
</netconf:rpc>

```

5. YANG Data Tree

```

module: ietf-te-kpi-telemetry
augment /te:te/te:tunnels/te:tunnel:
  +--rw te-scaling-intent
  |   +--rw scale-in-intent
  |   |   +--rw threshold-time?          uint32
  |   |   +--rw cooldown-time?          uint32
  |   |   +--rw scale-in-operation-type? scaling-criteria-operation
  |   |   +--rw scale-out-operation-type? scaling-criteria-operation
  |   |   +--rw scaling-condition* [performance-type]
  |   |   |   +--rw performance-type      identityref
  |   |   |   +--rw te-telemetry-tunnel-ref? -> /te:te/tunnels/tunnel/name
  |   +--rw scale-out-intent
  |   |   +--rw threshold-time?          uint32
  |   |   +--rw cooldown-time?          uint32
  |   |   +--rw scale-in-operation-type? scaling-criteria-operation
  |   |   +--rw scale-out-operation-type? scaling-criteria-operation
  |   |   +--rw scaling-condition* [performance-type]
  |   |   |   +--rw performance-type      identityref
  |   |   |   +--rw te-telemetry-tunnel-ref? -> /te:te/tunnels/tunnel/name
  +--ro te-telemetry
  |   +--ro id?                          string
  |   +--ro unidirectional-delay?        uint32
  |   +--ro unidirectional-min-delay?    uint32
  |   +--ro unidirectional-max-delay?    uint32
  |   +--ro unidirectional-delay-variation? uint32
  |   +--ro unidirectional-packet-loss?   decimal64
  |   +--ro unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro unidirectional-available-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro bidirectional-delay?          uint32
  |   +--ro bidirectional-min-delay?      uint32
  |   +--ro bidirectional-max-delay?      uint32
  |   +--ro bidirectional-delay-variation? uint32
  |   +--ro bidirectional-packet-loss?     decimal64
  |   +--ro bidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro bidirectional-available-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro bidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float
32
  |   +--ro utilized-percentage?          uint8
  |   +--ro te-ref?                      -> /te:te/tunnels/tunnel/name

module: ietf-actn-te-kpi-telemetry
augment /vn:actn/vn:vn/vn:vn-list:
  +--rw vn-scaling-intent
  |   +--rw scale-in-intent
  |   |   +--rw threshold-time?          uint32
  |   |   +--rw cooldown-time?          uint32
  |   |   +--rw scale-in-operation-type? scaling-criteria-operation
  |   |   +--rw scale-out-operation-type? scaling-criteria-operation
  |   |   +--rw scaling-condition* [performance-type]
  |   |   |   +--rw performance-type      identityref
  |   |   |   +--rw te-telemetry-tunnel-ref? -> /te:te/tunnels/tunnel/name
  |   +--rw scale-out-intent
  |   |   +--rw threshold-time?          uint32
  |   |   +--rw cooldown-time?          uint32

```



```

    |      +--rw scale-in-operation-type?      scaling-criteria-operation
    |      +--rw scale-out-operation-type?     scaling-criteria-operation
    |      +--rw scaling-condition* [performance-type]
    |          +--rw performance-type          identityref
    |          +--rw te-telemetry-tunnel-ref?  -> /te:te/tunnels/tunnel/name
+--ro vn-telemetry
  +--ro unidirectional-delay?                  uint32
  +--ro unidirectional-min-delay?              uint32
  +--ro unidirectional-max-delay?              uint32
  +--ro unidirectional-delay-variation?        uint32
  +--ro unidirectional-packet-loss?            decimal64
  +--ro unidirectional-residual-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro unidirectional-available-bandwidth?    rt-types:bandwidth-ieee-float
32
  +--ro unidirectional-utilized-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-delay?                  uint32
  +--ro bidirectional-min-delay?              uint32
  +--ro bidirectional-max-delay?              uint32
  +--ro bidirectional-delay-variation?        uint32
  +--ro bidirectional-packet-loss?            decimal64
  +--ro bidirectional-residual-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-available-bandwidth?    rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-utilized-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro utilized-percentage?                  uint8
  +--ro grouping-operation?                  grouping-operation
augment /vn:actn/vn:vn/vn:vn-list/vn:vn-member-list:
+--ro vn-member-telemetry
  +--ro unidirectional-delay?                  uint32
  +--ro unidirectional-min-delay?              uint32
  +--ro unidirectional-max-delay?              uint32
  +--ro unidirectional-delay-variation?        uint32
  +--ro unidirectional-packet-loss?            decimal64
  +--ro unidirectional-residual-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro unidirectional-available-bandwidth?    rt-types:bandwidth-ieee-float
32
  +--ro unidirectional-utilized-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-delay?                  uint32
  +--ro bidirectional-min-delay?              uint32
  +--ro bidirectional-max-delay?              uint32
  +--ro bidirectional-delay-variation?        uint32
  +--ro bidirectional-packet-loss?            decimal64
  +--ro bidirectional-residual-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-available-bandwidth?    rt-types:bandwidth-ieee-float
32
  +--ro bidirectional-utilized-bandwidth?     rt-types:bandwidth-ieee-float
32
  +--ro utilized-percentage?                  uint8
  +--ro te-grouped-params*                    -> /te:te/tunnels/tunnel/te-k
pi:te-
telemetry/id
  +--ro grouping-operation?                  grouping-operation

```

6. Yang Data Model

6.1. ietf-te-kpi-telemetry model

The YANG code is as follows:

```
<CODE BEGINS> file "ietf-te-kpi-telemetry@2018-07-02.yang"

module ietf-te-kpi-telemetry {

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

    prefix "te-tel";

    import ietf-te {
        prefix "te";
    }

    import ietf-te-types {
        prefix "te-types";
    }

    import ietf-routing-types {
        prefix "rt-types";
    }

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

    contact
        "Editor: Young Lee <leeyoung@huawei.com>
        Editor: Dhruv Dhody <dhruv.ietf@gmail.com>
        Editor: Ricard Vilalta <ricard.vilalta@cttc.es>
        Editor: Satish Karunanithi <satish.karunanithi@gmail.com>";

    description
        "This module describes telemetry for teas tunnel model";

    revision 2018-07-02 {
        description
            "Initial revision. This YANG file defines
            the reusable base types for TE telemetry.";
        reference
            "Derived from earlier versions of base YANG files";
    }

    /*
     * Identities
     */

    identity telemetry-param-type {
        description
```

```
        "Base identity for telemetry param types";
    }

    identity one-way-delay {
        base telemetry-param-type;
        description
            "To specify average Delay in one (forward)
            direction";
    }

    identity bidirectional-delay {
        base telemetry-param-type;
        description
            "To specify average Delay in both (forward and reverse)
            directions";
    }

    identity one-way-delay-variation {
        base telemetry-param-type;
        description
            "To specify average Delay Variation in one (forward) direction";
    }

    identity bidirectional-delay-variation {
        base telemetry-param-type;
        description
            "To specify average Delay Variation in both (forward and reverse)
            directions";
    }

    identity one-way-packet-loss {
        base telemetry-param-type;
        description
            "To specify packet loss in one (forward) direction.";
    }

    identity bidirectional-packet-loss {
        base telemetry-param-type;
        description
            "To specify packet loss in in both (forward and reverse)
            directions";
    }

    identity utilized-bandwidth {
        base telemetry-param-type;
        description
            "To specify utilized bandwidth over the specified source
            and destination.";
```

```
}

identity utilized-percentage {
    base telemetry-param-type;
    description
        "To specify utilization percentage of the entity
        (e.g., tunnel, link, etc.)";
}
/*
 * Enums
 */
typedef scaling-criteria-operation {
    type enumeration {
        enum AND {
            description
                "AND operation";
        }
        enum OR {
            description
                "OR operation";
        }
    }
    description
        "Operations to analyze list of scaling criterias";
}

/*
 * Groupings
 */

grouping bidirectional-telemetry-data {
    description
        "list all bidirectional telemetry data in this grouping";

    leaf bidirectional-delay {
        type uint32;
        units "microseconds";
        description
            "To specify average Delay in both (forward and reverse)
            directions during the measurement interval";
    }

    leaf bidirectional-min-delay {
        type uint32;
        units "microseconds";
        description
            "To specify minimum Delay in both (forward and reverse)";
    }
}
```

```
        directions during the measurement interval";
    }

    leaf bidirectional-max-delay {
        type uint32;
        units "microseconds";
        description
            "To specify maximum Delay in both (forward and reverse)
            directions during the measurement interval";
    }

    leaf bidirectional-delay-variation {
        type uint32;
        units "microseconds";
        description
            "To specify average Delay Variation in both
            (forward and reverse) directions during the
            measurement interval";
    }

    leaf bidirectional-packet-loss {
        type decimal64 {
            fraction-digits 4;
            range "0.0000..100.0000";
        }
        units "percent";
        description
            "To specify packet loss in in both (forward and reverse)
            directions";
    }

    leaf bidirectional-residual-bandwidth {
        type rt-types:bandwidth-ieee-float32;
        description
            "To specify residual bandwidth over the specified source
            and destination in bytes per seconds.";
        reference
            "RFC 3471";
    }

    leaf bidirectional-available-bandwidth {
        type rt-types:bandwidth-ieee-float32;
        description
            "To specify available bandwidth over the specified source
            and destination in bytes per seconds.";
        reference
            "RFC 3471";
    }

    leaf bidirectional-utilized-bandwidth {
        type rt-types:bandwidth-ieee-float32;
```

```
        description
            "To specify utilized bandwidth over the specified source
            and destination in bytes per seconds.";
        reference
            "RFC 3471";
    }
    leaf utilized-percentage {
        type uint8;
        units "percentage";
        description
            "integer indicating a percentage value (0..100) for
utilization";
    }
}

grouping scaling-duration {
    description
        "Base scaling criteria durations";
    leaf threshold-time {
        type uint32;
        units "seconds";
        description
            "The duration for which the criteria must hold true";
    }

    leaf cooldown-time {
        type uint32;
        units "seconds";
        description
            "The duration after a scaling-in/scaling-out action has been
            triggered, for which there will be no further operation";
    }
}

grouping scaling-criteria {
    description
        "Grouping for scaling criteria";
    leaf performance-type {
        type identityref {
            base telemetry-param-type;
        }
        description
            "Reference to the tunnel level telemetry type";
    }

    leaf te-telemetry-tunnel-ref {
        type leafref {
            path "/te:te/te:tunnels/te:tunnel/te:name";
        }
    }
}
```

```
        description
            "Reference to tunnel";
    }
}

grouping scaling-intent {
    description
        "Basic sclaing intent";

    uses scaling-duration;

    leaf scale-in-operation-type {
        type scaling-criteria-operation;
        default AND;
        description
            "Operation to be applied to check between scaling criterias to
            check if the scale in threshold condition has been met.
            Defaults to AND";
    }

    leaf scale-out-operation-type {
        type scaling-criteria-operation;
        default OR;
        description
            "Operation to be applied to check between scaling criterias to
            check if the scale out threshold condition has been met.
            Defaults to OR";
    }

    list scaling-condition {
        key "performance-type";
        description
            "Scaling conditions";
        uses scaling-criteria;
    }
}

/*
 * Augments
 */

augment "/te:te/te:tunnels/te:tunnel" {

    description
        "Augmentation parameters for config scaling-criteria
        TE tunnel topologies. Scale in/out criteria might be used
        for network autonomics in order the controller
        to react to a certain set of monitored params.";
```

```
    container te-scaling-intent {
      description
        "scaling intent";

      container scale-in-intent{
        description
          "scale-in";
        uses scaling-intent;
      }
      container scale-out-intent{
        description
          "scale-out";
        uses scaling-intent;
      }
    }
  container te-telemetry {
    config false;
    description
      "telemetry params";
    leaf id {
      type string;
      description "Id of telemetry param";
    }

    uses te-types:performance-metric-attributes;
    /* all unidirectional PM data is defined in this grouping */

    uses bidirectional-telemetry-data;
    /* all bidirectional PM data is defined in this grouping */

    leaf te-ref{
      type leafref{ path
        '/te:te/te:tunnels/te:tunnel/te:name'; }
      description "Reference to measured te tunnel";
    }
  }
}
```

<CODE ENDS>

6.2. ietf-actn-te-kpi-telemetry model

The YANG code is as follows:

```
<CODE BEGINS> file "ietf-actn-te-kpi-telemetry@2018-07-02.yang"

module ietf-actn-te-kpi-telemetry {

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

    prefix "actn-tel";

    import ietf-actn-vn {
        prefix "vn";
    }

    import ietf-te {
        prefix "te";
    }

    import ietf-te-types {
        prefix "te-types";
    }

    import ietf-te-kpi-telemetry {
        prefix "te-kpi";
    }

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

    contact
        "Editor: Young Lee <leeyoung@huawei.com>
        Editor: Dhruv Dhody <dhruv.ietf@gmail.com>
        Editor: Ricard Vilalta <ricard.vilalta@cttc.es>
        Editor: Satish Karunanithi <satish.karunanithi@gmail.com>";

    description
        "This module describes telemetry for actn vn model";

    revision 2018-07-02 {
        description
            "Initial revision. This YANG file defines
            the ACTN VN telemetry.";
        reference
```



```

        "Derived from earlier versions of base YANG files";
    }

    /*
     * Typedefs
     */

    typedef grouping-operation {

        type enumeration {
            enum MINIMUM { description "Select the minimum param"; }
            enum MAXIMUM { description "Select the maximum param"; }
            enum MEAN { description "Select the MEAN of the params"; }
            enum STD_DEV { description "Select the STD_DEV of the monitored para
ms"; }
            enum AND { description "Select the AND of the params"; }
            enum OR { description "Select the OR of the params"; }
        }
        description
            "Operations to analyze list of monitored params";
    }

    /*
     * Groupings
     */

    grouping vn-telemetry-param {
        description "augment of te-kpi:telemetry-param for VN specific params";

        leaf-list te-grouped-params {
            type leafref{
                path '/te:te/te:tunnels/te:tunnel/'+
                'te-kpi:te-telemetry/te-kpi:id';
            }
            description
                "Allows the definition of a vn-telemetry param
                as a grouping of underlying TE params";
        }

        leaf grouping-operation {
            type grouping-operation;
            description
                "describes the operation to apply to
                te-grouped-params";
        }
    }

```

```

    }

    /*
     * Augments
     */

    augment "/vn:actn/vn:vn/vn:vn-list" {

        description
            "Augmentation parameters for state TE VN topologies.";

        container vn-scaling-intent {
            description
                "scaling intent";

            container scale-in-intent{
                description
                    "VN scale-in";
                uses te-kpi:scaling-intent;
            }
            container scale-out-intent{
                description
                    "VN scale-out";
                uses te-kpi:scaling-intent;
            }
        }
        container vn-telemetry {
            config false;
            description
                "VN telemetry params";

            uses te-types:performance-metric-attributes;
            uses te-kpi:bidirectional-telemetry-data;
            leaf grouping-operation {
                type grouping-operation;
                description "describes the operation to apply to the VN-members"
            }
        }
    }

    /*
     * VN-member augment
     */
    augment "/vn:actn/vn:vn/vn:vn-list/vn:vn-member-list" {
        description
            "Augmentation parameters for state TE vn member topologies.";
        container vn-member-telemetry {

```

```
        config false;
    description
        "VN member telemetry params";

        uses te-types:performance-metric-attributes;
        uses te-kpi:bidirectional-telemetry-data;
        uses vn-telemetry-param;
    }
}
```

<CODE ENDS>

7. Security Considerations

The configuration, state, and action data defined in this document are designed to be accessed via a management protocol with a secure transport layer, such as NETCONF [RFC6241]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a preconfigured subset of all available NETCONF protocol operations and content.

A number of configuration data nodes defined in this document are writable/deletable (i.e., "config true") These data nodes may be considered sensitive or vulnerable in some network environments.

8. IANA Considerations

TDB

9. Acknowledgements

10. References

10.1. Informative References

[RFC4110] R. Callon and M. Suzuki, "A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC 4110, July 2005.

- [RFC6020] M. Bjorklund, Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.
- [Service-YANG] Q. Wu, W. Liu and A. Farrel, "Service Models Explained", draft-wu-opsawg-service-model-explained, work in progress.
- [Netmod-Yang-Model-Classification] D. Bogdanovic, B. Claise, and C. Moberg, "YANG Module Classification", draft-ietf-netmod-yang-model-classification, work in progress.
- [Netconf] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241.
- [Restconf] A. Bierman, M. Bjorklund, and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf, work in progress.
- [Routing-Types] X. Liu, et al, "Routing Area Common YANG Data Types", draft-ietf-rtgwg-routing-types, work in progress.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, March 2018,

10.2. Normative References

- [ACTN-Frame] D. Cecarelli and Y. Lee, "Framework for Abstraction and Control of Traffic Engineered Networks", draft-ietf-teas-actn-framework, work in progress.
- [TE-Topology] X. Liu, et al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress.
- [TE-Tunnel] T. Saad (Editor), "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te, work in progress.
- [ACTN-VN] Y. Lee (Editor), "A Yang Data Model for ACTN VN Operation", draft-lee-teas-actn-vn-yang, work in progress.

[L3SM-YANG] S. Litkowski, L.Tomotaki, and K. Ogaki, "YANG Data Model for L3VPN service delivery", draft-ietf-l3sm-l3vpn-service-model, work in progress.

[PCEP-Service-Aware] D. Dhody, et al., "Extensions to the Path Computation Element Communication Protocol (PCEP) to compute service aware Label Switched Path (LSP)", draft-ietf-pce-pcep-service-aware, work in progress.

[ACTN-PERF] 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-03, work in progress.

11. Contributors

Authors' Addresses

Young Lee
Huawei Technologies
5340 Legacy Drive Suite 173
Plano, TX 75024, USA

Email: leeyoung@huawei.com

Dhruv Dhody
Huawei Technology
Leela Palace
Bangalore, Karnataka 560008
India

Email: dhruv.dhody@huawei.com

Satish Karunanithi
Huawei Technology
Leela Palace
Bangalore, Karnataka 560008
India

Email: satish.karunanithi@gmail.com

Ricard Vilalta
Centre Tecnologic de Telecomunicacions de Catalunya (CTTC/CERCA)
Av. Carl Friedrich Gauss 7
08860 - Castelldefels
Barcelona (Spain)
Email: ricard.vilalta@cttc.es

Daniel King
Lancaster University

Email: d.king@lancaster.ac.uk

Daniele Ceccarelli
Ericsson
Torshamnsgatan, 48
Stockholm, Sweden

Email: daniele.ceccarelli@ericsson.com

TEAS WG
Internet Draft
Intended status: standard track
Expires: December 28, 2018

Young Lee
Dhruv Dhody
Huawei

Daniele Ceccarelli
Ericsson

Jeff Tantsura
Nuage

Giuseppe Fioccola
Telecom Italia

Qin Wu
Huawei

June 29, 2018

Traffic Engineering and Service Mapping Yang Model

draft-lee-teas-te-service-mapping-yang-09

Abstract

This document provides a YANG data model to map service model (e.g., L3SM) and Traffic Engineering model (e.g., TE Tunnel or ACTN VN model). This model is referred to as TE service Mapping Model. This model is applicable to the operation's need for a seamless control and management of their VPN services with TE tunnel support.

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/lid-abstracts.txt>

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

This Internet-Draft will expire on December 29, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction.....	3
2. TE-Service Mapping Model.....	4
2.1. VN/Tunnel Selection Requirements.....	5
2.2. Availability Requirements.....	6
3. L3VPN Architecture in ACTN context.....	6
4. YANG Data Tree.....	9
5. Yang Data Model.....	10
6. Security.....	17
7. IANA Considerations.....	18
8. Acknowledgements.....	18
9. References.....	18
9.1. Informative References.....	18
10. Contributors.....	19
Authors' Addresses.....	20

1. Introduction

Data models are a representation of objects that can be configured or monitored within a system. Within the IETF, YANG [RFC6020] is the language of choice for documenting data models, and YANG models have been produced to allow configuration or modeling of a variety of network devices, protocol instances, and network services. YANG data models have been classified in [Netmod-Yang-Model-Classification] and [RFC8309].

[RFC4110] provides a framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs). [L3SM-YANG] provides a L3VPN service delivery YANG model for PE-based VPNs. The scope of this draft is limited to a set of domain under the same network operators to deliver services requiring TE tunnels.

[ACTN-VN-YANG] describes how customers or end to end orchestrators can request and/or instantiate a generic virtual network service. [ACTN-Applicability] describes a connection between IETF YANG model classifications to ACTN interfaces. In particular, it describes the customer service model can be mapped into the CMI (CNC-MDSC Interface) of the ACTN architecture.

While IP/MPLS PNC is responsible for provisioning the VPN service on the PE nodes, the MDSC can coordinate how to map the VPN services with TE tunnels. This is consistent with the two of the core functions of the MDSC specified in [ACTN-Frame]:

- . Customer mapping/translation function: This function is to map customer requests/commands into network provisioning requests that can be sent to the Physical Network Controller (PNC) according to business policies provisioned statically or dynamically. Specifically, it provides mapping and translation of a customer's service request into a set of parameters that are specific to a network type and technology such that network configuration process is made possible.
- . Virtual service coordination function: This function translates customer service-related information into virtual network service operations in order to seamlessly operate virtual networks while meeting a customer's service requirements. In the context of ACTN, service/virtual service coordination includes a number of service orchestration functions such as multi-destination load balancing, guarantees of service quality, bandwidth and throughput. It also includes notifications for service fault and performance degradation and so forth.

The YANG model described in this document provides an ACTN TE-service mapping model that enables a seamless service mapping across L1/2/3 VPN, ACTN VN and TE-Topology/TE-tunnel models at the controllers.

2. TE-Service Mapping Model

The role of TE-service Mapping model is to create a mapping relationship between service models and TE models so that VN/VPN service instantiation would be seamlessly provided by the underlying TE networks. It also allows for the customers to access operational state information as to how their services are instantiated with the underlying TE topology or TE tunnels. This mapping will facilitate a seamless service management operation with underlay-TE network visibility.

Figure 1 shows TE-Service Mapping Model's scope.

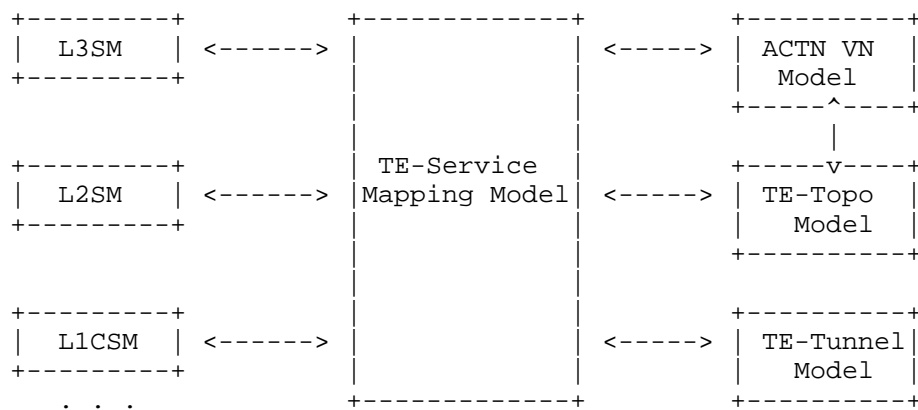


Figure 1. TE-Service Mapping

As seen in Figure 1, the TE-Service Mapping Model provides a mapping between the LxSM and the ACTN VN YANG model which allows customers (internal or external) to create a VN that meets the customer's service objective with various constraints via TE-topology model [TE-topo]. The model also supports a direct mapping between LxSM to TE-topology or TE-tunnel.

The TE-service model developed in this document can also be extended to support other services beyond L3SM, L2SM and L1CSM.

Moreover, the TE-Service Mapping model provides additional service parameters and policies that are not included in the respective service models such as L3SM [RFC8299], L2SM [L2SM-YANG] and L1CSM [L1CSM-YANG]. For example, how VN/TE tunnel should be created (e.g., with an isolation level) for a certain service instance is described in the TE-Service Mapping model.

2.1. VN/Tunnel Selection Requirements

In some cases, under the confines of service policy, TE network slicing isolation requirement may need to be supported for the VPN service. This may occur when there are no suitable existing TE tunnels that can support VPN service requirements. Or the operator would like to dynamically create and bind tunnels to the VPN, which could not be shared for network slicing.

To summarize there are three modes of VN/Tunnel selection operations to be supported, but not limited to:

- o New VN/Tunnel Binding - Customer could request an L3VPN service [L3SM-YANG] with a new VN/Tunnel not shared with other existing services. This is to meet VPN isolation requirement. Further the mapping yang model described in Section 5 of this document is used to set this mapping between the L3VPN service and the ACTN VN. Note that this could be done dynamically. The VN (and TE tunnels) could be bound to the L3VPN and not used for any other VPN.

Under this mode, the following sub-categories can be supported:

1. Hard Isolation with deterministic characteristics:
Customer would request an LxVPN service using a set of TE Tunnels with a deterministic characteristics requirement (e.g., no latency variation) and that cannot be not shared with other LxVPN services nor compete for bandwidth with other Tunnels.
2. Hard Isolation: This is similar to the above case without deterministic characteristics requirements.
3. Soft Isolation: Customer would request an LxVPN service using a set of MPLS-TE tunnel which cannot be shared with other LxVPN services.

- o VN/Tunnel Sharing - Customer could request an L3VPN service [L3SM-Yang], and with this model as input, the PNC configures the different network elements to deliver the service. Each network element would select a tunnel based on the configuration. With this mode, new tunnels (or VN) are not created for each VPN. Thus, the tunnels can be shared across multiple VPN. Further the mapping yang model described in Section 5 of this document is used to get the mapping between the L3VPN and the tunnels in use. No modification is allowed when an existing tunnel is selected.
- o VN/Tunnel Modify - This mode allows the modification of the properties of the existing VN/tunnel (e.g., bandwidth) when VN/Tunnel Selection Mode is applied.

Other mode of VN/Tunnel selection operations could be easily added to the current model in the future.

2.2. Availability Requirement

Availability is another LxVPN's service requirement or intent that needs to be conveyed by the TE & Service Mapping model. Availability is a probabilistic measure of the length of time a VPN/VN or network slice instance is functioning. The following level SHOULD be supported.

- o 99.9999%
- o 99.999%
- o 99.99%
- o 99.9%
- o 99%

The availability level will need to be translated into network specific policy such as protection/reroute policy associated with a VN or Tunnel, which is not in the scope of this draft.

3. L3VPN Architecture in ACTN context

Figure 1 shows the architectural context of this document.

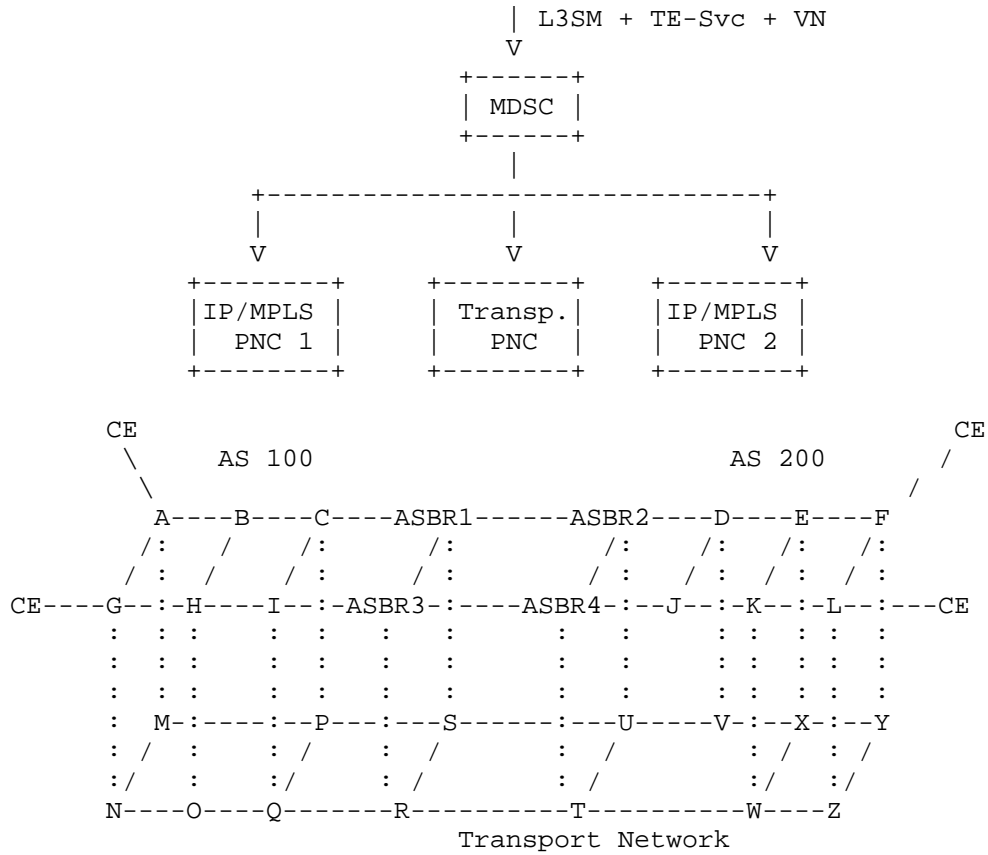


Figure 1: L3VPN Architecture from the IP+Optical Network Perspective

There are three main entities in the architecture.

- . MDSC: This entity is responsible for coordinating a L3VPN service request (expressed in L3SM) with the IP PNC and the Transport PNC. One of the key responsibilities of the MDSC for TE services is to coordinate with both the IP PNC and the Transport PNC for the mapping of L3VPN Service Model and ACTN VN model. With the VN/TE-tunnel binding case, the MDSC will need to coordinate with the Transport PNC to dynamically create the TE-tunnel(s) in the Transport network as needed. These tunnels are added as links in

the IP Layer topology. The MDSC coordinates with IP PNC to create the TE-tunnel(s) in the IP layer, as part of the ACTN VN creation.

- . IP/MPLS PNC: This entity is responsible for device configuration to create PE-PE L3VPN tunnels for the VPN customer and for the configuration of the L3VPN VRF on the PE nodes. Each network element would select a tunnel based on the configuration.
- . Transport PNC: This entity is responsible for device configuration for TE tunnels in the transport networks.

High-Level Control Flows

1. Customer asks for a L3VPN between CE1 and CE2 with TE constraints using L3SM model. The customer can provide tunnel creation policy where it allows dynamic VN/TE tunnel creation or not. Under this policy, dynamic VN/TE tunnels can be created when there are no proper VN/TE-tunnels that can support L3VPN tunnels or when there is a strict isolation requirement for the VPN service, e.g., no sharing with other tunnels is allowed.
2. The MDSC determines if it needs to create a new VN, and if that is the case, ACTN VN YANG [ACTN-VN-YANG] is used to configure a new VN based on this VPN and map the VPN service to ACTN VN. In case an existing tunnel is to be used, each device will select which tunnel to use and populates this mapping information.
3. The MDSC interacts with both the IP/MPLS PNC and the Transport PNC to create a PE-PE tunnel in the IP network mapped to a TE tunnel in the transport network by providing the inter-layer access points and tunnel requirements. The specific service information are passed to the IP/MPLS PNC for the actual VPN configuration and activation.
 - a. The Transport PNC creates the corresponding TE tunnel matching with the access point and egress point.
 - b. The IP/MPLS PNC maps the VPN ID with the corresponding TE tunnel ID to bind these two IDs.
4. The IP/MPLS PNC creates/updates a VRF instance for this VPN customer. This is not in the scope of this document..

4. YANG Data Tree

```

module: ietf-te-service-mapping
  +--rw te-service-mapping
    +--rw service-mapping
      +--rw mapping-list* [map-id]
        +--rw map-id          uint32
        +--rw map-type?      map-type
        +--rw (service)?
          +--:(l3vpn)
            | +--rw l3vpn-ref?      -> /l3:l3vpn-svc/vpn-services/vpn-
service/vpn-id
          +--:(l2vpn)
            | +--rw l2vpn-ref?      -> /l2:l2vpn-svc/vpn-services/vpn-
service/vpn-id
          +--:(l1vpn)
            | +--rw l1vpn-ref?      -> /l1:l1cs/service/service-
list/subscriber-l1vc-id
          +--rw (te)?
            +--:(actn-vn)
              | +--rw actn-vn-ref?  -> /vn:actn/vn/vn-list/vn-id
            +--:(te-topo)
              | +--rw vn-topology-id? te-types:te-topology-id
              | +--rw abstract-node? -> /nw:networks/network/node/node-
id
            +--:(te-tunnel)
              +--rw te-tunnel-list* te:tunnel-ref
      +--rw site-mapping
        +--rw mapping-list* [map-id]
          +--rw map-id          uint32
          +--rw (service)?
            +--:(l3vpn)
              | +--rw l3vpn-ref?    -> /l3:l3vpn-svc/sites/site/site-id
            +--:(l2vpn)
              | +--rw l2vpn-ref?    -> /l2:l2vpn-svc/sites/site/site-id
            +--:(l1vpn)
              | +--rw l1vpn-ref?    -> /l1:l1cs/access/uni-list/UNI-ID
          +--rw (te)?
            +--:(actn-vn)
              | +--rw actn-vn-ref?  -> /vn:actn/ap/access-point-
list/access-point-id
            +--:(te)
              +--rw ltp?            te-types:te-tp-id

```


5. Yang Data Model

The YANG code is as follows:

<CODE BEGINS> file "ietf-te-service-mapping@2018-04-05.yang"

```
module ietf-te-service-mapping {  
  
    namespace "urn:ietf:params:xml:ns:yang:ietf-te-service-mapping";  
  
    prefix "tm";  
  
    import ietf-l3vpn-svc {  
        prefix "l3";  
    }  
  
    import ietf-l2vpn-svc {  
        prefix "l2";  
    }  
  
    import ietf-l1csm {  
        prefix "l1";  
    }  
  
    import ietf-te-types {  
        prefix "te-types";  
    }  
  
    import ietf-network {  
        prefix "nw";  
    }  
  
    import ietf-te {  
        prefix "te";  
    }  
  
    import ietf-actn-vn {  
        prefix "vn";  
    }  
  
    organization
```

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

contact
  "Editor: Young Lee <leeyoung@huawei.com>
    Dhruv Dhody <dhruv.ietf@gmail.com>";
description
  "This module contains a YANG module for the mapping of
  service (e.g. L3VPN) to the TE tunnels or ACTN VN.";

revision 2018-04-05 {
  description
    "initial version.";
  reference
    "TBD";
}

/*
 * Identities
 */
identity service-type {
  description
    "Base identity from which specific service types are
    derived.";
}

identity l3vpn-service {
  base service-type;
  description
    "L3VPN service type.";
}

identity l2vpn-service {
  base service-type;
  description
    "L2VPN service type.";
}

identity l1vpn-service {
  base service-type;
  description
    "L1VPN connectivity service type.";
}
/*
 * Enum
```

```

    */
    typedef map-type {
        type enumeration {
            enum "new" {
                description
                    "The new VN/tunnels are binded to the service";
            }
            enum "select" {
                description
                    "The VPN service selects an existing tunnel with no
modification";
            }
            enum "modify" {
                description
                    "The VPN service selects an existing tunnel and allows
to modify the properties of the tunnel (e.g., b/w) ";
            }
        }
        description
            "The map-type";
    }

    /*
    * Groupings
    */
    grouping service-ref{
        description
            "The reference to the service.";
        choice service {
            description
                "The service";
            case l3vpn {
                leaf l3vpn-ref {
                    type leafref {
                        path "/l3:l3vpn-svc/l3:vpn-services/"
                        + "l3:vpn-service/l3:vpn-id";
                    }
                    description
                        "The reference to L3VPN Service Yang Model";
                }
            }
            case l2vpn {
                leaf l2vpn-ref {
                    type leafref {
                        path "/l2:l2vpn-svc/l2:vpn-services/"

```

```

        + "l2:vpn-service/l2:vpn-id";
    }
    description
        "The reference to L2VPN Service Yang Model";
}

}
case l1vpn {
    leaf l1vpn-ref {
        type leafref {
            path "/l1:llcs/l1:service/"
                + "l1:service-list/l1:subscriber-l1vc-id";
        }
        description
            "The reference to L1VPN Service Yang Model";
    }
}

}

grouping site-ref {
    description
        "The reference to the site.";
    choice service {
        description
            "The service choice";
        case l3vpn {
            leaf l3vpn-ref {
                type leafref {
                    path "/l3:l3vpn-svc/l3:sites/l3:site/"
                        + "l3:site-id";
                }
                description
                    "The reference to L3VPN Service Yang Model";
            }
        }
        case l2vpn {
            leaf l2vpn-ref {
                type leafref {
                    path "/l2:l2vpn-svc/l2:sites/l2:site/"
                        + "l2:site-id";
                }
                description
                    "The reference to L2VPN Service Yang Model";
            }
        }
    }
}

```

```

    }
  }
  case llvpn {
    leaf llvpn-ref {
      type leafref {
        path "/l1:llcs/l1:access/l1:uni-list/"
          + "l1:UNI-ID";
      }
      description
        "The reference to L1VPN Connectivity Service Yang
Model";
    }
  }
}

grouping te-ref {
  description
    "The reference to TE.";
  choice te {
    description
      "The TE";
    case actn-vn {
      leaf actn-vn-ref {
        type leafref {
          path "/vn:actn/vn:vn/vn:vn-list/vn:vn-id";
        }
        description
          "The reference to ACTN VN";
      }
    }
    case te-topo {
      leaf vn-topology-id {
        type te-types:te-topology-id;
        description
          "An identifier to the TE Topology Model
          where the abstract nodes and links of
          the Topology can be found for Type 2
          VNS";
      }
      leaf abstract-node {
        type leafref {

```

```

        path "/nw:networks/nw:network/nw:node/"
        + "nw:node-id";
    }
    description
        "a reference to the abstract node in TE
        Topology";
    }
}
case te-tunnel {
    leaf-list te-tunnel-list {
        type te:tunnel-ref;
        description
            "Reference to TE Tunnels";
    }
}
}
}

grouping te-endpoint-ref {
    description
        "The reference to TE endpoints.";
    choice te {
        description
            "The TE";
        case actn-vn {
            leaf actn-vn-ref {
                type leafref {
                    path "/vn:actn/vn:ap/vn:access-point-list"
                    + "/vn:access-point-id";
                }
                description
                    "The reference to ACTN VN";
            }
        }
        case te {
            leaf ltp {
                type te-types:te-tp-id;
                description
                    "Reference LTP in the TE-topology";
            }
        }
    }
}
}

```

```
}
grouping service-mapping {
  description
    "Mapping between Services and TE";
  container service-mapping {
    description
      "Mapping between Services and TE";

    list mapping-list {
      key "map-id";
      description
        "Mapping identified via a map-id";
      leaf map-id {
        type uint32;
        description
          "a unique mapping identifier";
      }
      leaf map-type {
        type map-type;
        description
          "Tunnel Bind or Tunnel Selection";
      }
      uses service-ref;

      uses te-ref;
    }
  }
}
grouping site-mapping {
  description
    "Mapping between VPN access site and TE
    endpoints or AP";
  container site-mapping {
    description
      "Mapping between VPN access site and TE
      endpoints or AP";
    list mapping-list {
      key "map-id";
      description
        "Mapping identified via a map-id";
      leaf map-id {
        type uint32;
        description
          "a unique mapping identifier";
      }
    }
  }
}
```

```
        "a unique mapping identifier";
    }
    uses site-ref;

    uses te-endpoint-ref;
}

}

/*
 * Configuration data nodes
 */
container te-service-mapping {
    description
        "Mapping between Services and TE";

    uses service-mapping;

    uses site-mapping;
}

}
```

<CODE ENDS>

6. Security

The configuration, state, and action data defined in this document are designed to be accessed via a management protocol with a secure transport layer, such as NETCONF [RFC6241]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a preconfigured subset of all available NETCONF protocol operations and content.

A number of configuration data nodes defined in this document are writable/deletable (i.e., "config true") These data nodes may be considered sensitive or vulnerable in some network environments.

7. IANA Considerations

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

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

This document registers the following YANG modules in the YANG Module

Names registry [RFC7950]:

```
-----  
name:          ietf-te-service-mapping  
namespace:     urn:ietf:params:xml:ns:yang:ietf-te-service-mapping  
reference:     RFC XXXX (TDB)  
-----
```

8. Acknowledgements

We thank Diego Caviglia and Igor Bryskin for useful discussions and motivation for this work.

9. References

9.1. Informative References

- [RFC4110] R. Callon and M. Suzuki, "A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC 4110, July 2005.
- [RFC6020] M. Bjorklund, Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

- [RFC8309] Q. Wu, W. Liu and A. Farrel, "Service Models Explained", RFC 8309, January 2018.
- [Netmod-Yang-Model-Classification] D. Bogdanovic, B. Claise, and C. Moberg, "YANG Module Classification", draft-ietf-netmod-yang-model-classification, work in progress.
- [Netconf] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241.
- [Restconf] A. Bierman, M. Bjorklund, and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf, work in progress.
- [ACTN-Frame] D. Cecarelli and Y. Lee, "Framework for Abstraction and Control of Traffic Engineered Networks", draft-ietf-teas-actn-framework, work in progress.
- [TE-Topology] X. Liu, et. al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress.
- [TE-Tunnel] T. Saad (Editor), "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te, work in progress.
- [ACTN-VN-YANG] Y. Lee (Editor), "A Yang Data Model for ACTN VN Operation", draft-lee-teas-actn-vn-yang, work in progress.
- [RFC8299] Q. Wu, S. Litkowski, L. Tomotaki, and K. Ogaki, "YANG Data Model for L3VPN service delivery", RFC 8299, January 2018.
- [L2SM-YANG] B. Wen, et al, "A YANG Data Model for L2VPN Service Delivery", draft-ietf-l2sm-l2vpn-service-model, work in progress.
- [L1CSM-YANG] G. Fioccola, et al, "A Yang Data Model for L1 Connectivity Service Model (L1CSM)", draft-ietf-ccamp-l1csm-yang, work in progress.

10. Contributors

Authors' Addresses

Young Lee
Huawei Technologies
5340 Legacy Drive
Plano, TX 75023, USA
Phone: (469)277-5838

Email: leeyoung@huawei.com

Dhruv Dhody
Huawei Technologies

Email: dhruv.ietf@gmail.com

Daniele Ceccarelli
Ericsson
Torshamnsgatan, 48
Stockholm, Sweden

Email: daniele.ceccarelli@ericsson.com

Jeff Tantsura
Huawei

Email: jefftant@gmail.com

Giuseppe Fioccola
Telecom Italia
Email: giuseppe.fioccola@telecomitalia.it

Qin Wu
Huawei
Email: bill.wu@huawei.com

