

TEAS Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2017

D. Ceccarelli
Ericsson
L. Berger
LabN Consulting, L.L.C.
October 27, 2016

Generalized Routing Interface Switching Capability Descriptor Switching
Capability Specific Information
draft-ceccarelli-teas-gneralized-scsi-00

Abstract

This document defines a generic information structure for information carried in routing protocol Interface Switching Capability Descriptor (ISCD) Switching Capability Specific Information (SCSI) fields. This "Generalized SCSI" can be used with routing protocols that define GMPLS ISCDs, and any specific technology. This document does not modify an existing technology specific formats and is defined for use in conjunction with new GMPLS Switching Capability types.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 30, 2017.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Generalized SCSI Formats	3
4. Procedures	4
5. Security Considerations	5
6. IANA Considerations	5
7. References	6
7.1. Normative References	6
7.2. Informative References	6
Authors' Addresses	7

1. Introduction

The Interface Switching Capability Descriptor (ISCD) [RFC4202] allows routing protocols such as OSPF and ISIS to carry technology specific information in the the Switching Capability-specific information (SCSI) field, see [RFC4203] and [RFC5307]. The format of an SCSI field is dictated by the specific technology being represented as indicated by the ISCD Switching Capability (SC) type field. Existing Switching Capabilities are managed by IANA in the Switching Types registry and the related "IANA-GMPLS-TC-MIB" definitions.

[RFC7138] introduced a "sub-TLV" structure to its technology specific SCSI field. The Sub-Type-Length-Value (TLV) based approach allows for greater flexibility in the structure, ordering, and ability to support extensions of the SC (technology) specific format. This Sub-TLV approach is also used in [RFC7688].

This document generalizes this approach and defines a new generalized SCSI field format for use by future specific technologies and Switching Capability types. The generalized SCSI carries SCSI-TLVs that may be defined within the scope of a specific technology, or shared across multiple technologies (e.g., [I-D.ietf-ccamp-ospf-availability-extension]). This document also establishes a registry for SCSI-TLV definitions that may be shared across multiple technologies.

2. Terminology

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

3. Generalized SCSI Formats

The Generalized SCSI is composed of zero or more variable length type-length-value fields which are each called a SCSI-TLV. There are no specific size restrictions on these SCSI-TLV. Size and other formatting restrictions may be imposed by the routing protocol ISCD field, refer to [RFC4203] and [RFC5307].

The SCSI-TLV format is:

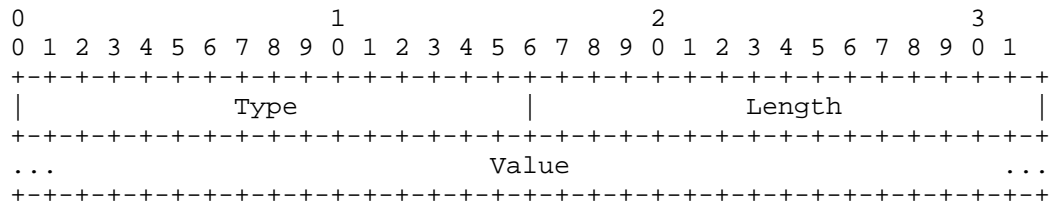


Figure 1: TLV format

Type (2 octets):

This field indicates the type and structure of the information contained in the Value field. Note that the value range of this field has been split in two. The lower range is used to indicated technology specific formats, while the higher range is reserved for formats that can be used with more than one technology. See Section 6

Length (2 octets):

This field MUST be set to the size, in octets (bytes), of the Value field. The value of the field MUST be zero or divisible by 4. Note that this implies that the Value field can be omitted or contain padding.

Value (variable):

A variable length field, formatted according to the value of the Type field. This field can be omitted for certain types.

4. Procedures

The Generalized SCSI is used with ISCDs (defined in [RFC4203] and [RFC5307]) of technologies whose Switching Capability definition reference this document. The corollary of this is that the Generalized SCSI MUST NOT be used for ISCDs of technologies whose Switching Capability definition do not reference this document.

The Generalized SCSI MAY contain a sequence of zero or more SCSI-TLVs. Sub-TLV parsing (format) errors, such as an underrun or overrun, MUST be treated as a malformed ISCD. SCSI-TLVs MUST be processed in the order received and, if re-originated, ordering MUST be preserved. Unknown SCSI-TLVs MUST be ignored and transparently processed, i.e., re-originated when appropriate. Processing related to multiple SCSI-TLVs of the same type may be further refined based

on the definition on the type.

5. Security Considerations

This document does not introduce any security issue beyond those discussed in [RFC4203] and [RFC5307]. As discussed there, the information carried in ISCDs are not used for SPF computation or normal routing and the extensions here defined do not have direct effect on IP routing. Tampering with GMPLS TE LSAs may have an effect on the underlying transport network. Mechanisms such as [RFC2154] and [RFC5304] to protect the transmission of this information are suggested.

6. IANA Considerations

This document defines a new SCSI-TLV that is carried in the SCSI field of the ISCDs defined in [RFC4203] and [RFC5307]. The SCSI-TLV includes a 16-bit type identifier (the Type field). The same Type field values are applicable to the new SCSI-TLV.

IANA is requested to create and maintain a new registry, the "Generalized SCSI (Switching Capability Specific Information) TLVs Types" registry under either the the "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Parameters" registry or a new "Generalized Multi-Protocol Label Switching (GMPLS) Routing Parameters" registry.

The definition of the new registry is as follows:

Value	SCSI-TLV	SwitchCap	Reference
-----	-----	-----	-----
0	Reserved		[This ID]
1-32768	Unassigned, for use by specific technology	[per value]	[This ID]
32768-65535	Unassigned, for others	(Any, or value list)	[This ID]

New allocation requests to this registry SHALL indicate the value or values to be used in the SwitchCap column.

The registry should be established with registration policies of "Standards Action" for Standards Track documents, and "Specification Required" for other documents, see [RFC5226]. The designated expert

is any current TEAS WG chair.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC4202] Kompella, K., Ed. and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, DOI 10.17487/RFC4202, October 2005, <<http://www.rfc-editor.org/info/rfc4202>>.
- [RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <<http://www.rfc-editor.org/info/rfc4203>>.
- [RFC5307] Kompella, K., Ed. and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, <<http://www.rfc-editor.org/info/rfc5307>>.

7.2. Informative References

- [I-D.ietf-ccamp-ospf-availability-extension] Long, H., Ye, M., Mirsky, G., D'Alessandro, A., and H. Shah, "OSPF-TE Link Availability Extension for Links with Variable Discrete Bandwidth", draft-ietf-ccamp-ospf-availability-extension-08 (work in progress), October 2016.
- [RFC2154] Murphy, S., Badger, M., and B. Wellington, "OSPF with Digital Signatures", RFC 2154, DOI 10.17487/RFC2154, June 1997, <<http://www.rfc-editor.org/info/rfc2154>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <<http://www.rfc-editor.org/info/rfc5304>>.

- [RFC7138] Ceccarelli, D., Ed., Zhang, F., Belotti, S., Rao, R., and J. Drake, "Traffic Engineering Extensions to OSPF for GMPLS Control of Evolving G.709 Optical Transport Networks", RFC 7138, DOI 10.17487/RFC7138, March 2014, <<http://www.rfc-editor.org/info/rfc7138>>.
- [RFC7688] Lee, Y., Ed. and G. Bernstein, Ed., "GMPLS OSPF Enhancement for Signal and Network Element Compatibility for Wavelength Switched Optical Networks", RFC 7688, DOI 10.17487/RFC7688, November 2015, <<http://www.rfc-editor.org/info/rfc7688>>.

Authors' Addresses

Daniele Ceccarelli
Ericsson
Torshamnsgatan 21
Kista - Stockholm
Sweden

Email: daniele.ceccarelli@ericsson.com

Lou Berger
LabN Consulting, L.L.C.

TEAS Working Group
Internet Draft
Intended status: Informational
Expires: November 28, 2018

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

May 28, 2018

Framework for Abstraction and Control of Traffic Engineered Networks
draft-ietf-teas-actn-framework-15

Abstract

Traffic Engineered networks have a variety of mechanisms to facilitate the separation of the data plane and control plane. They also have a range of management and provisioning protocols to configure and activate network resources. These mechanisms represent key technologies for enabling flexible and dynamic networking. The term "Traffic Engineered network" refers to a network that uses any connection-oriented technology under the control of a distributed or centralized control plane to support dynamic provisioning of end-to-end connectivity.

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

This document provides a framework for Abstraction and Control of Traffic Engineered Networks (ACTN) to support virtual network services and connectivity services.

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 November 3, 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. Overview.....	4
2.1. Terminology.....	5
2.2. VNS Model of ACTN.....	7
2.2.1. Customers.....	9
2.2.2. Service Providers.....	10
2.2.3. Network Operators.....	10
3. ACTN Base Architecture.....	10
3.1. Customer Network Controller.....	12
3.2. Multi-Domain Service Coordinator.....	13
3.3. Provisioning Network Controller.....	13
3.4. ACTN Interfaces.....	14
4. Advanced ACTN Architectures.....	15
4.1. MDSC Hierarchy.....	15
4.2. Functional Split of MDSC Functions in Orchestrators.....	16
5. Topology Abstraction Methods.....	17
5.1. Abstraction Factors.....	17
5.2. Abstraction Types.....	18
5.2.1. Native/White Topology.....	18
5.2.2. Black Topology.....	19
5.2.3. Grey Topology.....	20
5.3. Methods of Building Grey Topologies.....	21

5.3.1. Automatic Generation of Abstract Topology by Configuration.....	21
5.3.2. On-demand Generation of Supplementary Topology via Path Compute Request/Reply.....	21
5.4. Hierarchical Topology Abstraction Example.....	22
5.5. VN Recursion with Network Layers.....	24
6. Access Points and Virtual Network Access Points.....	25
6.1. Dual-Homing Scenario.....	27
7. Advanced ACTN Application: Multi-Destination Service.....	28
7.1. Pre-Planned End Point Migration.....	29
7.2. On the Fly End-Point Migration.....	30
8. Manageability Considerations.....	30
8.1. Policy.....	31
8.2. Policy Applied to the Customer Network Controller.....	32
8.3. Policy Applied to the Multi-Domain Service Coordinator...	32
8.4. Policy Applied to the Provisioning Network Controller....	32
9. Security Considerations.....	33
9.1. CNC-MDSC Interface (CMI).....	34
9.2. MDSC-PNC Interface (MPI).....	34
10. IANA Considerations.....	34
11. References.....	35
11.1. Informative References.....	35
12. Contributors.....	36
Authors' Addresses.....	37
APPENDIX A - Example of MDSC and PNC Functions Integrated in A Service/Network Orchestrator.....	37

1. Introduction

The term "Traffic Engineered network" refers to a network that uses any connection-oriented technology under the control of a distributed or centralized control plane to support dynamic provisioning of end-to-end connectivity. Traffic Engineered (TE) networks have a variety of mechanisms to facilitate the separation of data plane and control plane including distributed signaling for path setup and protection, centralized path computation for planning and traffic engineering, and a range of management and provisioning protocols to configure and activate network resources. These mechanisms represent key technologies for enabling flexible and dynamic networking. Some examples of networks that are in scope of this definition are optical networks, Multiprotocol Label Switching (MPLS) Transport Profile (MPLS-TP) networks [RFC5654], and MPLS-TE networks [RFC2702].

One of the main drivers for Software Defined Networking (SDN) [RFC7149] is a decoupling of the network control plane from the data

plane. This separation has been achieved for TE networks with the development of MPLS/GMPLS [RFC3945] and the Path Computation Element (PCE) [RFC4655]. One of the advantages of SDN is its logically centralized control regime that allows a global view of the underlying networks. Centralized control in SDN helps improve network resource utilization compared with distributed network control. For TE-based networks, a PCE may serve as a logically centralized path computation function.

This document describes a set of management and control functions used to operate one or more TE networks to construct virtual networks that can be presented to customers and that are built from abstractions of the underlying TE networks. For example, a link in the customer's network is constructed from a path or collection of paths in the underlying networks. We call this set of functions "Abstraction and Control of Traffic Engineered Networks" (ACTN).

2. Overview

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

- . Separation of service requests from service delivery so that the configuration and operation of a network is transparent from the point of view of the customer, but remains responsive to the customer's services and business needs.
- . Network abstraction: As described in [RFC7926], abstraction is the process of applying policy to a set of information about a TE network to produce selective information that represents the potential ability to connect across the network. The process of abstraction presents the connectivity graph in a way that is independent of the underlying network technologies, capabilities, and topology so that the graph can be used to plan and deliver network services in a uniform way
- . Coordination of resources across multiple independent networks and multiple technology layers to provide end-to-end services regardless of whether the networks use SDN or not.

As networks evolve, the need to provide support for distinct services, separated service orchestration, and resource abstraction have emerged as key requirements for operators. In order to support multiple customers each with its own view of and control of the server network, a network operator needs to partition (or "slice") or manage sharing of the network resources. Network slices can be assigned to each customer for guaranteed usage which is a step further than shared use of common network resources.

Furthermore, each network represented to a customer can be built from virtualization of the underlying networks so that, for example, a link in the customer's network is constructed from a path or collection of paths in the underlying network.

ACTN can facilitate virtual network operation via the creation of a single virtualized network or a seamless service. This supports operators in viewing and controlling different domains (at any dimension: applied technology, administrative zones, or vendor-specific technology islands) and presenting virtualized networks to their customers.

The ACTN framework described in this document facilitates:

- . Abstraction of the underlying network resources to higher-layer applications and customers [RFC7926].
- . Virtualization of particular underlying resources, whose selection criterion is the allocation of those resources to a particular customer, application, or service [ONF-ARCH].
- . TE Network slicing of infrastructure to meet specific customers' service requirements.
- . Creation of an abstract environment allowing operators to view and control multi-domain networks as a single abstract network.
- . The presentation to customers of networks as a virtual network via open and programmable interfaces.

2.1. Terminology

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

- . Domain: A domain [RFC4655] is any collection of network elements within a common sphere of address management or path computation responsibility. Specifically within this document we mean a part of an operator's network that is under common management (i.e., under shared operational management using the same instances of a tool and the same policies). Network elements will often be grouped into domains based on technology types, vendor profiles, and geographic proximity.
- . Abstraction: This process is defined in [RFC7926].

- . TE Network Slicing: In the context of ACTN, a TE network slice is a collection of resources that is used to establish a logically dedicated virtual network over one or more TE networks. TE network slicing allows a network operator to provide dedicated virtual networks for applications/customers over a common network infrastructure. The logically dedicated resources are a part of the larger common network infrastructures that are shared among various TE network slice instances which are the end-to-end realization of TE network slicing, consisting of the combination of physically or logically dedicated resources.
- . Node: A node is a vertex on the graph representation of a TE topology. In a physical network topology, a node corresponds to a physical network element (NE) such as a router. In an abstract network topology, a node (sometimes called an abstract node) is a representation as a single vertex of one or more physical NEs and their connecting physical connections. The concept of a node represents the ability to connect from any access to the node (a link end) to any other access to that node, although "limited cross-connect capabilities" may also be defined to restrict this functionality. Network abstraction may be applied recursively, so a node in one topology may be created by applying abstraction to the nodes in the underlying topology.
- . Link: A link is an edge on the graph representation of a TE topology. Two nodes connected by a link are said to be "adjacent" in the TE topology. In a physical network topology, a link corresponds to a physical connection. In an abstract network topology, a link (sometimes called an abstract link) is a representation of the potential to connect a pair of points with certain TE parameters (see [RFC7926] for details). Network abstraction may be applied recursively, so a link in one topology may be created by applying abstraction to the links in the underlying topology.
- . Abstract Topology: The topology of abstract nodes and abstract links presented through the process of abstraction by a lower layer network for use by a higher layer network.
- . A Virtual Network (VN) is a network provided by a service provider to a customer for the customer to use in any way it wants as though it was a physical network. There are two views of a VN as follows:

- a) The VN can be abstracted as a set of edge-to-edge links (a Type 1 VN). Each link is referred 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.
- b) The VN can also be abstracted as a topology of virtual nodes and virtual links (a Type 2 VN). The operator needs to map the VN to actual resource assignment, which is known as virtual network embedding. The nodes in this case include physical end points, border nodes, and internal nodes as well as abstracted nodes. Similarly the links include physical access links, inter-domain links, and intra-domain links as well as abstract links.

Clearly a Type 1 VN is a special case of a Type 2 VN.

- . Access link: A link between a customer node and a operator node.
- . Inter-domain link: A link between domains under distinct management administration.
- . Access Point (AP): An AP is a logical identifier shared between the customer and the operator used to identify an access link. The AP is used by the customer when requesting a VNS. Note that the term "TE Link Termination Point" (LTP) defined in [TE-Topo] describes the end points of links, while an AP is a common identifier for the link itself.
- . VN Access Point (VNAP): A VNAP is the binding between an AP and a given VN.
- . Server Network: As defined in [RFC7926], a server network is a network that provides connectivity for another network (the Client Network) in a client-server relationship.

2.2. VNS Model of ACTN

A Virtual Network Service (VNS) is the service agreement between a customer and operator to provide a VN. When a VN is a simple connectivity between two points, the difference between VNS and connectivity service becomes blurred. There are three types of VNS defined in this document.

- o Type 1 VNS refers to a VNS in which the customer is allowed to create and operate a Type 1 VN.
- o Type 2a and 2b VNS refer to VNSs in which the customer is allowed to create and operates a Type 2 VN. With a Type 2a VNS, the VN is statically created at service configuration time and the customer is not allowed to change the topology (e.g., by adding or deleting abstract nodes and links). A Type 2b VNS is the same as a Type 2a VNS except that the customer is allowed to make dynamic changes to the initial topology created at service configuration time.

VN Operations are functions that a customer can exercise on a VN depending on the agreement between the customer and the operator.

- o VN Creation allows a customer to request the instantiation of a VN. This could be through off-line pre-configuration or through dynamic requests specifying attributes to a Service Level Agreement (SLA) to satisfy the customer's objectives.
- o Dynamic Operations allow a customer to modify or delete the VN. The customer can further act upon the virtual network to create/modify/delete virtual links and nodes. These changes will result in subsequent tunnel management in the operator's networks.

There are three key entities in the ACTN VNS model:

- Customers
- Service Providers
- Network Operators

These entities are related in a three tier model as shown in Figure 1.

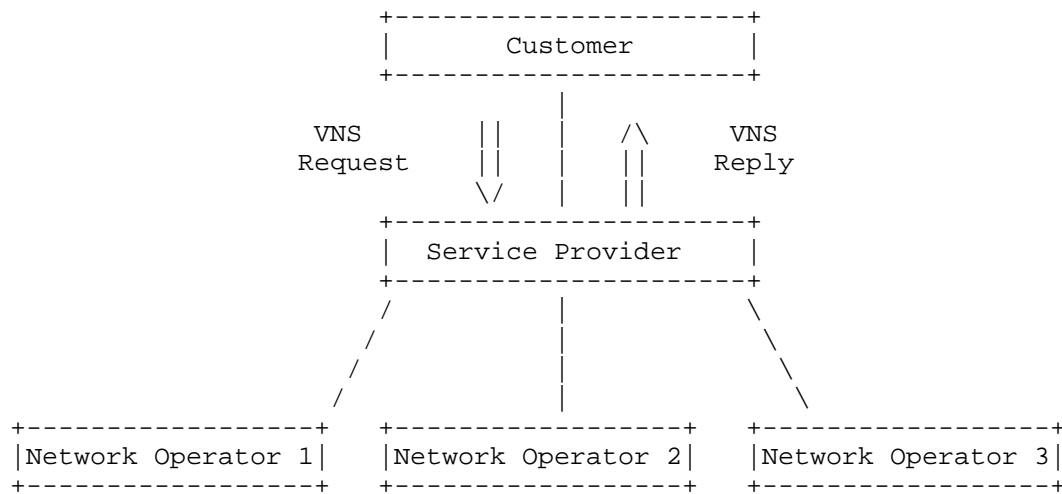


Figure 1: The Three Tier Model.

The commercial roles of these entities are described in the following sections.

2.2.1. Customers

Basic customers include fixed residential users, mobile users, and small enterprises. Each requires a small amount of resources and is characterized by steady requests (relatively time invariant). Basic customers do not modify their services themselves: if a service change is needed, it is performed by the provider as a proxy.

Advanced customers include enterprises and governments. Such customers ask for both point-to point and multipoint connectivity with high resource demands varying significantly in time. This is one of the reasons why a bundled service offering is not enough and it is desirable to provide each advanced customer with a customized virtual network service. Advanced customers may also have the ability to modify their service parameters within the scope of their virtualized environments. The primary focus of ACTN is Advanced Customers.

As customers are geographically spread over multiple network operator domains, they have to interface to multiple operators and may have to support multiple virtual network services with different underlying objectives set by the network operators. To enable these

customers to support flexible and dynamic applications they need to control their allocated virtual network resources in a dynamic fashion, and that means that they need a view of the topology that spans all of the network operators. Customers of a given service provider can in turn offer a service to other customers in a recursive way.

2.2.2. Service Providers

In the scope of ACTN, service providers deliver VNSs to their customers. Service providers may or may not own physical network resources (i.e., may or may not be network operators as described in Section 2.2.3). When a service provider is the same as the network operator, this is similar to existing VPN models applied to a single operator although it may be hard to use this approach when the customer spans multiple independent network operator domains.

When network operators supply only infrastructure, while distinct service providers interface to the customers, the service providers are themselves customers of the network infrastructure operators. One service provider may need to keep multiple independent network operators because its end-users span geographically across multiple network operator domains. In some cases, service provider is also a network operator when it owns network infrastructure on which service is provided.

2.2.3. Network Operators

Network operators are the infrastructure operators that provision the network resources and provide network resources to their customers. The layered model described in this architecture separates the concerns of network operators and customers, with service providers acting as aggregators of customer requests.

3. ACTN Base Architecture

This section provides a high-level model of ACTN showing the interfaces and the flow of control between components.

The ACTN architecture is based on a 3-tier reference model and allows for hierarchy and recursion. The main functionalities within an ACTN system are:

- . Multi-domain coordination: This function oversees the specific aspects of different domains and builds a single abstracted end-to-end network topology in order to coordinate end-to-end path computation and path/service provisioning. Domain

sequence path calculation/determination is also a part of this function.

- . Abstraction: This function provides an abstracted view of the underlying network resources for use by the customer - a customer may be the client or a higher level controller entity. This function includes network path computation based on customer service connectivity request constraints, path computation based on the global network-wide abstracted topology, and the creation of an abstracted view of network resources allocated to each customer. These operations depend on customer-specific network objective functions and customer traffic profiles.
- . Customer mapping/translation: This function is to map customer requests/commands into network provisioning requests that can be sent from the Multi-Domain Service Coordinator (MDSC) to the Provisioning Network Controller (PNC) according to business policies provisioned statically or dynamically at the Operations Support System (OSS)/ Network Management System (NMS). 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: 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 base ACTN architecture defines three controller types and the corresponding interfaces between these controllers. The following types of controller are shown in Figure 2:

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

Figure 2 also shows the following interfaces:

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

. SBI - Southbound Interface

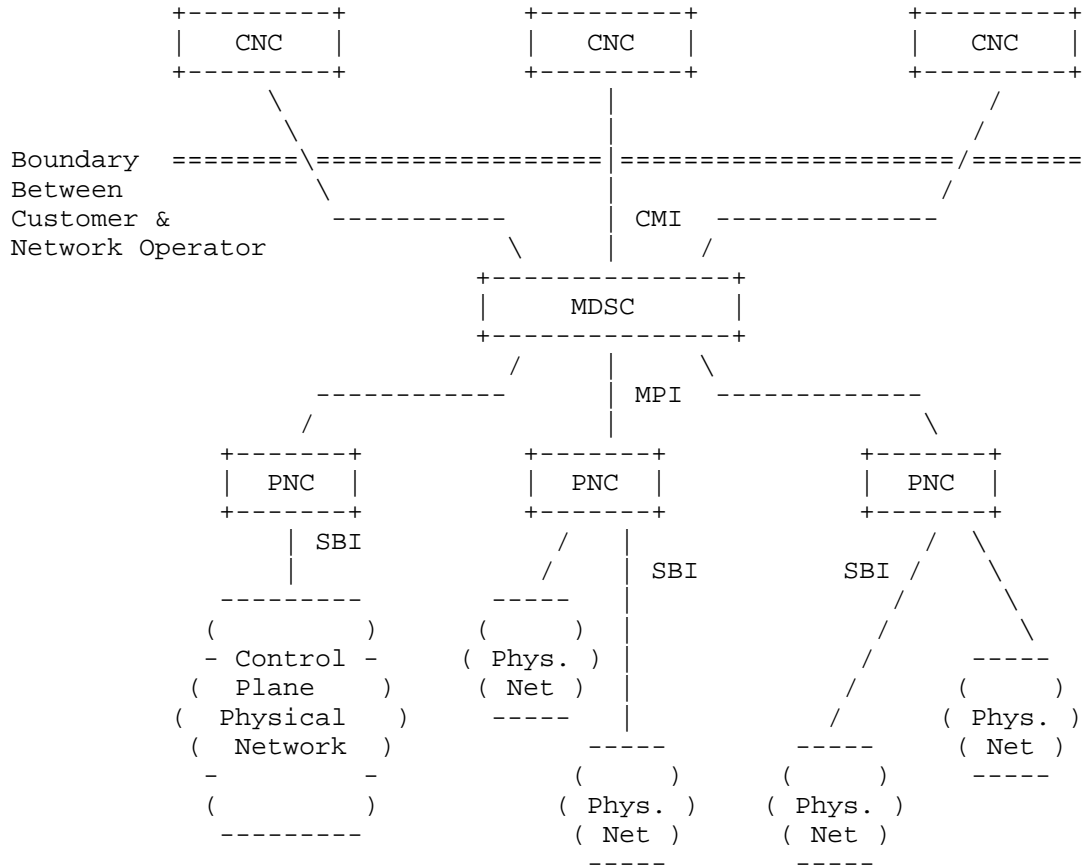


Figure 2: ACTN Base Architecture

Note that this is a functional architecture: an implementation and deployment might collocate one or more of the functional components. Figure 2 shows a case where service provider is also a network operator.

3.1. Customer Network Controller

A Customer Network Controller (CNC) is responsible for communicating a customer's VNS requirements to the network operator over the CNC-MDSC Interface (CMI). It has knowledge of the end-points associated

with the VNS (expressed as APs), the service policy, and other QoS information related to the service.

As the Customer Network Controller directly interfaces to the applications, it understands multiple application requirements and their service needs. The capability of a CNC beyond its CMI role is outside the scope of ACTN and may be implemented in different ways. For example, the CNC may in fact be a controller or part of a controller in the customer's domain, or the CNC functionality could also be implemented as part of a service provider's portal.

3.2. Multi-Domain Service Coordinator

A Multi-Domain Service Coordinator (MDSC) is a functional block that implements all of the ACTN functions listed in Section 3 and described further in Section 4.2. Two functions of the MDSC, namely, multi-domain coordination and virtualization/abstraction are referred to as network-related functions while the other two functions, namely, customer mapping/translation and virtual service coordination are referred to as service-related functions. The MDSC sits at the center of the ACTN model between the CNC that issues connectivity requests and the Provisioning Network Controllers (PNCs) that manage the network resources.

The key point of the MDSC (and of the whole ACTN framework) is detaching the network and service control from underlying technology to help the customer express the network as desired by business needs. The MDSC envelopes the instantiation of the right technology and network control to meet business criteria. In essence it controls and manages the primitives to achieve functionalities as desired by the CNC.

In order to allow for multi-domain coordination a 1:N relationship must be allowed between MDSCs and PNCs.

In addition to that, it could also be possible to have an M:1 relationship between MDSCs and PNC to allow for network resource partitioning/sharing among different customers not necessarily connected to the same MDSC (e.g., different service providers) but all using the resources of a common network infrastructure operator.

3.3. Provisioning Network Controller

The Provisioning Network Controller (PNC) oversees configuring the network elements, monitoring the topology (physical or virtual) of

the network, and collecting information about the topology (either raw or abstracted).

The PNC functions can be implemented as part of an SDN domain controller, a Network Management System (NMS), an Element Management System (EMS), an active PCE-based controller [Centralized] or any other means to dynamically control a set of nodes and implementing a north bound interface from the standpoint of the nodes (which is out of the scope of this document). A PNC domain includes all the resources under the control of a single PNC. It can be composed of different routing domains and administrative domains, and the resources may come from different layers. The interconnection between PNC domains is illustrated in Figure 3.

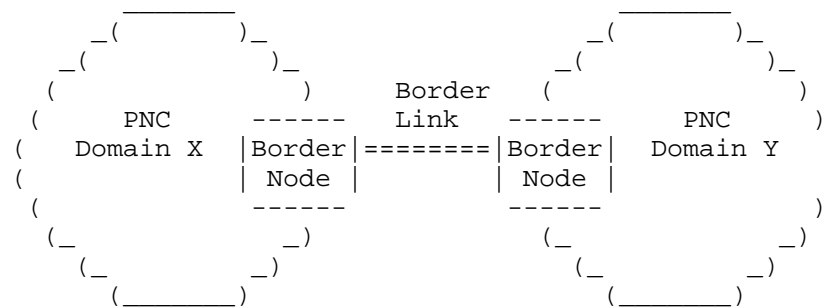


Figure 3: PNC Domain Borders

3.4. ACTN Interfaces

Direct customer control of transport network elements and virtualized services is not a viable proposition for network operators due to security and policy concerns. Therefore, the network has to provide open, programmable interfaces, through which customer applications can create, replace and modify virtual network resources and services in an interactive, flexible and dynamic fashion.

Three interfaces exist in the ACTN architecture as shown in Figure 2.

- . CMI: The CNC-MDSC Interface (CMI) is an interface between a CNC and an MDSC. The CMI is a business boundary between customer and network operator. It is used to request a VNS for an application. All service-related information is conveyed over this interface (such as the VNS type, topology, bandwidth, and

service constraints). Most of the information over this interface is agnostic of the technology used by network operators, but there are some cases (e.g., access link configuration) where it is necessary to specify technology-specific details.

- . MPI: The MDSC-PNC Interface (MPI) is an interface between an MDSC and a PNC. It communicates requests for new connectivity or for bandwidth changes in the physical network. In multi-domain environments, the MDSC needs to communicate with multiple PNCs each responsible for control of a domain. The MPI presents an abstracted topology to the MDSC hiding technology specific aspects of the network and hiding topology according to policy.
- . SBI: The Southbound Interface (SBI) is out of scope of ACTN. Many different SBIs have been defined for different environments, technologies, standards organizations, and vendors. It is shown in Figure 3 for reference reason only.

4. Advanced ACTN Architectures

This section describes advanced configurations of the ACTN architecture.

4.1. MDSC Hierarchy

A hierarchy of MDSCs can be foreseen for many reasons, among which are scalability, administrative choices, or putting together different layers and technologies in the network. In the case where there is a hierarchy of MDSCs, we introduce the terms higher-level MDSC (MDSC-H) and lower-level MDSC (MDSC-L). The interface between them is a recursion of the MPI. An implementation of an MDSC-H makes provisioning requests as normal using the MPI, but an MDSC-L must be able to receive requests as normal at the CMI and also at the MPI. The hierarchy of MDSCs can be seen in Figure 4.

Another implementation choice could foresee the usage of an MDSC-L for all the PNCs related to a given technology (e.g., Internet Protocol (IP)/Multiprotocol Label Switching (MPLS)) and a different MDSC-L for the PNCs related to another technology (e.g., Optical Transport Network (OTN)/Wavelength Division Multiplexing (WDM)) and an MDSC-H to coordinate them.

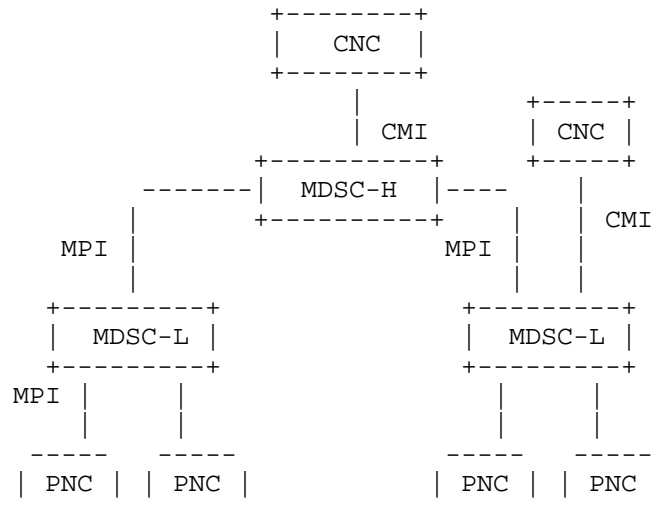
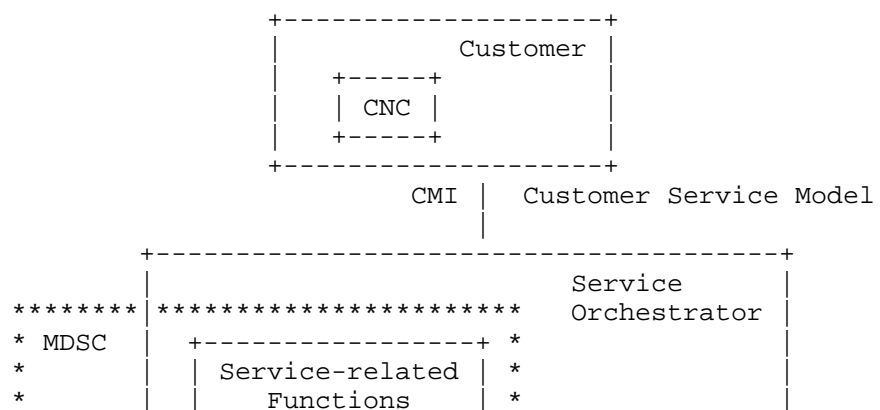


Figure 4: MDSC Hierarchy

The hierarchy of MDSC can be recursive, where an MDSC-H is in turn an MDSC-L to a higher level MDSC-H.

4.2. Functional Split of MDSC Functions in Orchestrators

An implementation choice could separate the MDSC functions into two groups, one group for service-related functions and the other for network-related functions. This enables the implementation of a service orchestrator that provides the service-related functions of the MDSC and a network orchestrator that provides the network-related functions of the MDSC. This split is consistent with the Yet Another Next Generation (YANG) service model architecture described in [Service-YANG]. Figure 5 depicts this and shows how the ACTN interfaces may map to YANG models.



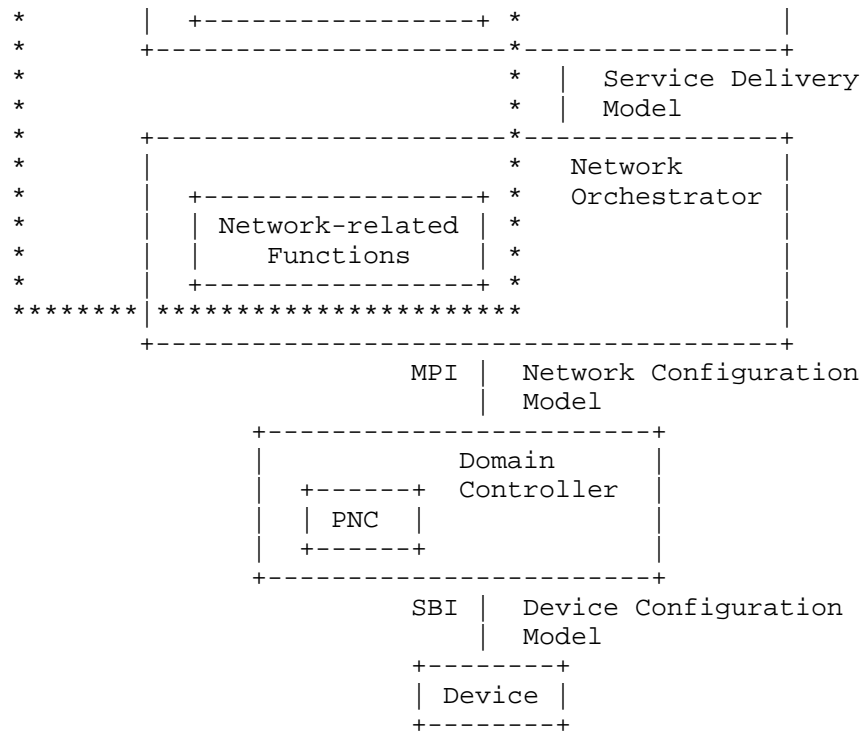


Figure 5: ACTN Architecture in the Context of the YANG Service Models

5. Topology Abstraction Methods

Topology abstraction is described in [RFC7926]. This section discusses topology abstraction factors, types, and their context in the ACTN architecture.

Abstraction in ACTN is performed by the PNC when presenting available topology to the MDSC, or by an MDSC-L when presenting topology to an MDSC-H. This function is different to the creation of a VN (and particularly a Type 2 VN) which is not abstraction but construction of virtual resources.

5.1. Abstraction Factors

As discussed in [RFC7926], abstraction is tied with policy of the networks. For instance, per an operational policy, the PNC would not provide any technology specific details (e.g., optical parameters for Wavelength Switched Optical Network (WSO) in the abstract topology it provides to the MDSC. Similarly, policy of the

networks may determine the abstraction type as described in Section 5.2.

There are many factors that may impact the choice of abstraction:

- Abstraction depends on the nature of the underlying domain networks. For instance, packet networks may be abstracted with fine granularity while abstraction of optical networks depends on the switching units (such as wavelengths) and the end-to-end continuity and cross-connect limitations within the network.
- Abstraction also depends on the capability of the PNCs. As abstraction requires hiding details of the underlying network resources, the PNC's capability to run algorithms impacts the feasibility of abstraction. Some PNC may not have the ability to abstract native topology while other PNCs may have the ability to use sophisticated algorithms.
- Abstraction is a tool that can improve scalability. Where the native network resource information is of large size there is a specific scaling benefit to abstraction.
- The proper abstraction level may depend on the frequency of topology updates and vice versa.
- The nature of the MDSC's support for technology-specific parameters impacts the degree/level of abstraction. If the MDSC is not capable of handling such parameters then a higher level of abstraction is needed.
- In some cases, the PNC is required to hide key internal topological data from the MDSC. Such confidentiality can be achieved through abstraction.

5.2. Abstraction Types

This section defines the following three types of topology abstraction:

- . Native/White Topology (Section 5.2.1)
- . Black Topology (Section 5.2.2)
- . Grey Topology (Section 5.2.3)

5.2.1. Native/White Topology

This is a case where the PNC provides the actual network topology to the MDSC without any hiding or filtering of information, i.e., no

abstraction is performed. In this case, the MDSC has the full knowledge of the underlying network topology and can operate on it directly.

5.2.2. Black Topology

A black topology replaces a full network with a minimal representation of the edge-to-edge topology without disclosing any node internal connectivity information. The entire domain network may be abstracted as a single abstract node with the network's access/egress links appearing as the ports to the abstract node and the implication that any port can be 'cross-connected' to any other. Figure 6 depicts a native topology with the corresponding black topology with one virtual node and inter-domain links. In this case, the MDSC has to make a provisioning request to the PNCs to establish the port-to-port connection. If there is a large number of inter-connected domains, this abstraction method may impose a heavy coordination load at the MDSC level in order to find an optimal end-to-end path since the abstraction hides so much information that it is not possible to determine whether an end-to-end path is feasible without asking each PNC to set up each path fragment. For this reason, the MPI might need to be enhanced to allow the PNCs to be queried for the practicality and characteristics of paths across the abstract node.

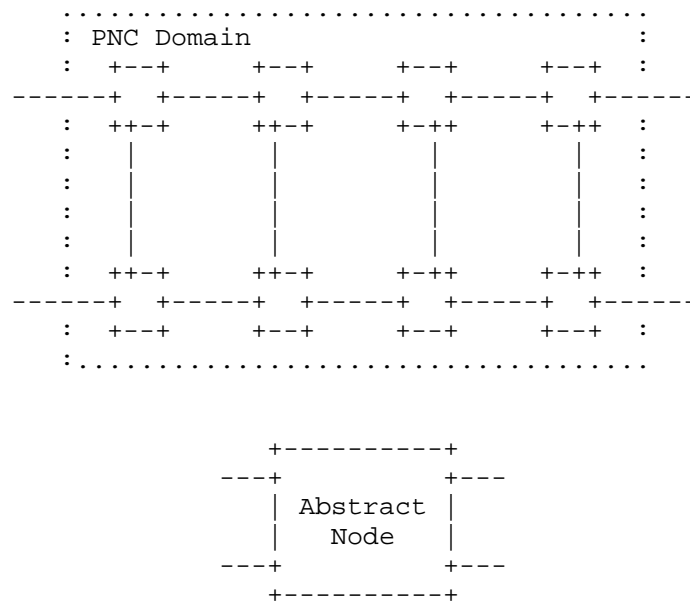


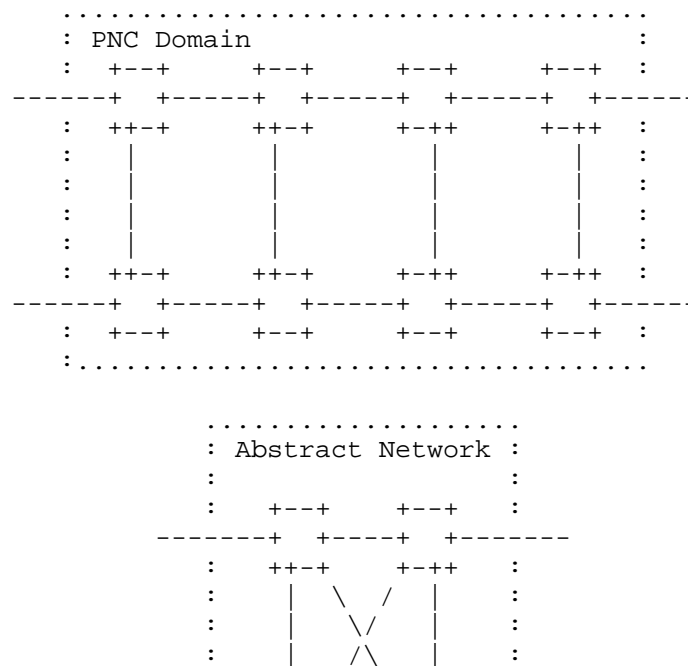
Figure 6: Native Topology with Corresponding Black Topology Expressed as an Abstract Node

5.2.3. Grey Topology

A grey topology represents a compromise between black and white topologies from a granularity point of view. In this case, the PNC exposes an abstract topology containing all PNC domains border nodes and an abstraction of the connectivity between those border nodes. This abstraction may contain either physical or abstract nodes/links.

Two types of grey topology are identified:

- . In a type A grey topology, border nodes are connected by a full mesh of TE links (see Figure 7).
- . In a type B grey topology, border nodes are connected over a more detailed network comprising internal abstract nodes and abstracted links. This mode of abstraction supplies the MDSC with more information about the internals of the PNC domain and allows it to make more informed choices about how to route connectivity over the underlying network.



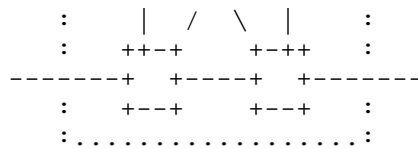


Figure 7: Native Topology with Corresponding Grey Topology

5.3. Methods of Building Grey Topologies

This section discusses two different methods of building a grey topology:

- . Automatic generation of abstract topology by configuration (Section 5.3.1)
- . On-demand generation of supplementary topology via path computation request/reply (Section 5.3.2)

5.3.1. Automatic Generation of Abstract Topology by Configuration

Automatic generation is based on the abstraction/summarization of the whole domain by the PNC and its advertisement on the MPI. The level of abstraction can be decided based on PNC configuration parameters (e.g., "provide the potential connectivity between any PE and any ASBR in an MPLS-TE network").

Note that the configuration parameters for this abstract topology can include available bandwidth, latency, or any combination of defined parameters. How to generate such information is beyond the scope of this document.

This abstract topology may need to be periodically or incrementally updated when there is a change in the underlying network or the use of the network resources that make connectivity more or less available.

5.3.2. On-demand Generation of Supplementary Topology via Path Compute Request/Reply

While abstract topology is generated and updated automatically by configuration as explained in Section 5.3.1, additional supplementary topology may be obtained by the MDSC via a path compute request/reply mechanism.

The abstract topology advertisements from PNCs give the MDSC the border node/link information for each domain. Under this scenario,

when the MDSC needs to create a new VN, the MDSC can issue path computation requests to PNCs with constraints matching the VN request as described in [ACTN-YANG]. An example is provided in Figure 8, where the MDSC is creating a P2P VN between AP1 and AP2. The MDSC could use two different inter-domain links to get from domain X to domain Y, but in order to choose the best end-to-end path it needs to know what domain X and Y can offer in terms of connectivity and constraints between the PE nodes and the border nodes.

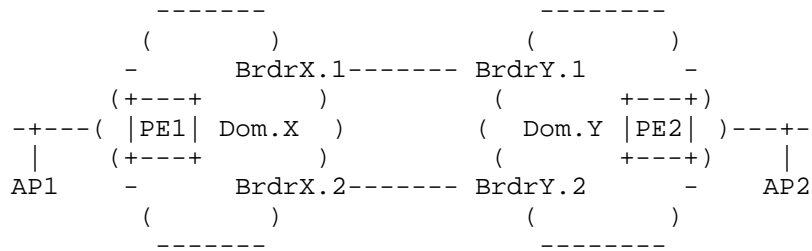
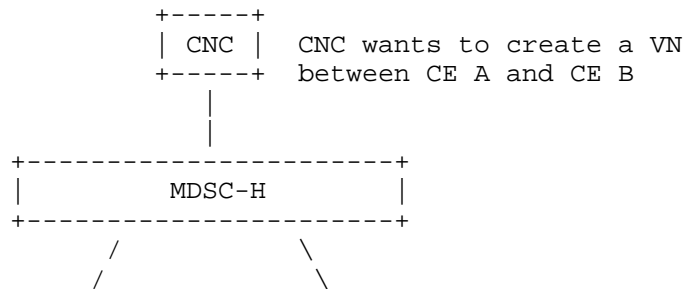


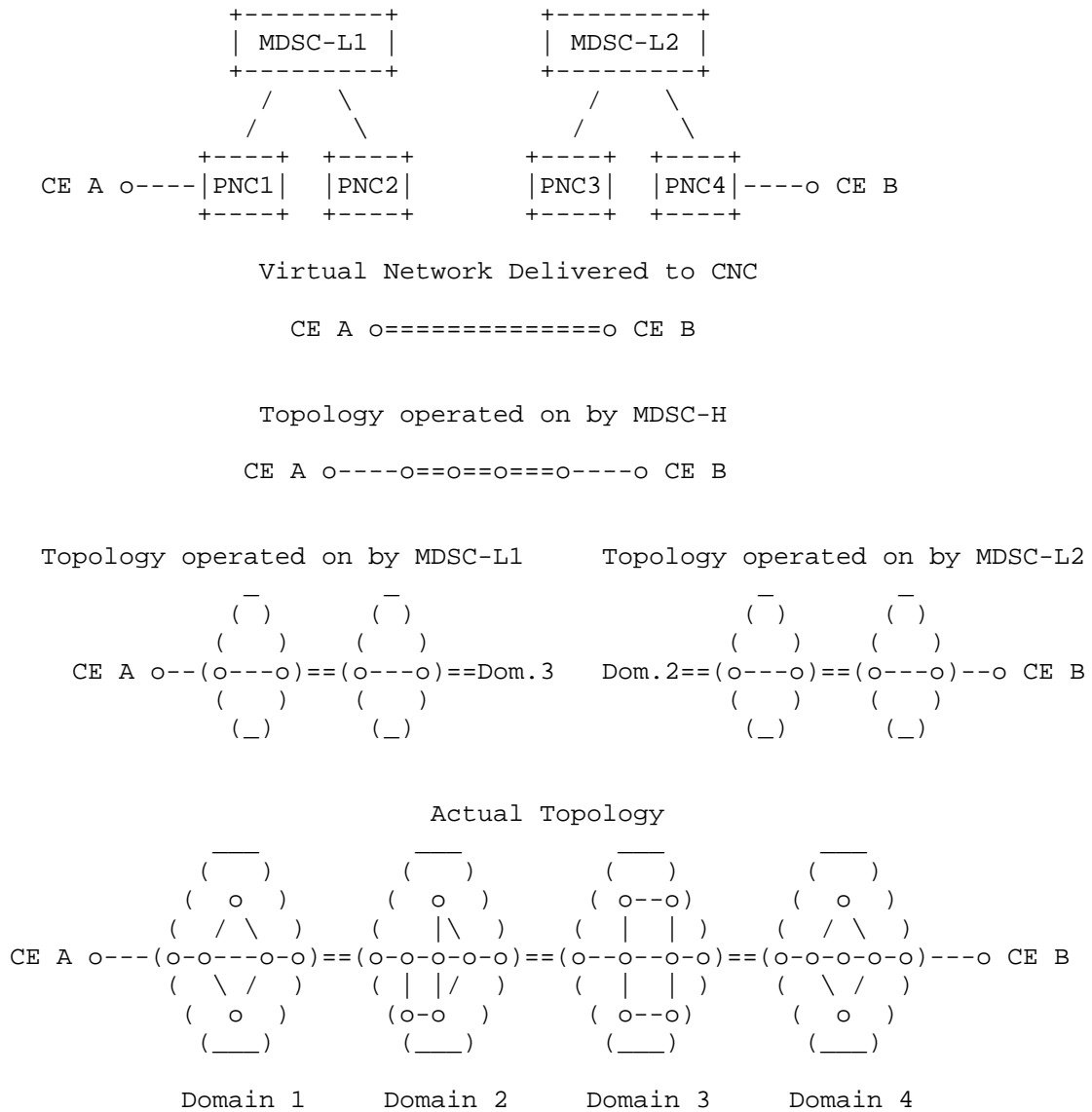
Figure 8: A Multi-Domain Example

The MDSC issues a path computation request to PNC.X asking for potential connectivity between PE1 and border node BrdrX.1 and between PE1 and BrdrX.2 with related objective functions and TE metric constraints. A similar request for connectivity from the border nodes in domain Y to PE2 will be issued to PNC.Y. The MDSC merges the results to compute the optimal end-to-end path including the inter domain links. The MDSC can use the result of this computation to request the PNCs to provision the underlying networks, and the MDSC can then use the end-to-end path as a virtual link in the VN it delivers to the customer.

5.4. Hierarchical Topology Abstraction Example

This section illustrates how topology abstraction operates in different levels of a hierarchy of MDSCs as shown in Figure 9.





Where

o is a node
 --- is a link
 === border link

Figure 9: Illustration of Hierarchical Topology Abstraction

In the example depicted in Figure 9, there are four domains under control of PNCs PNC1, PNC2, PNC3, and PNC4. MDSC-L1 controls PNC1

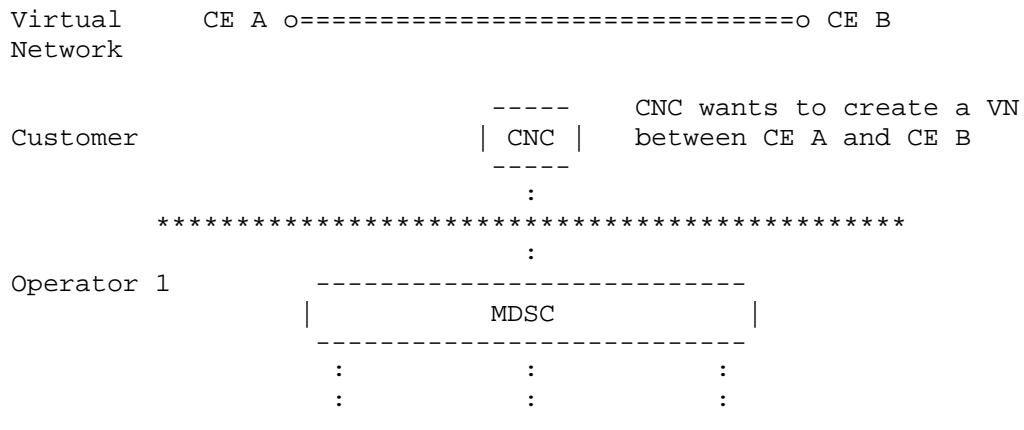
and PNC2 while MDSC-L2 controls PNC3 and PNC4. Each of the PNCs provides a grey topology abstraction that presents only border nodes and links across and outside the domain. The abstract topology MDSC-L1 that operates is a combination of the two topologies from PNC1 and PNC2. Likewise, the abstract topology that MDSC-L2 operates is shown in Figure 9. Both MDSC-L1 and MDSC-L2 provide a black topology abstraction to MDSC-H in which each PNC domain is presented as a single virtual node. MDSC-H combines these two topologies to create the abstraction topology on which it operates. MDSC-H sees the whole four domain networks as four virtual nodes connected via virtual links.

5.5. VN Recursion with Network Layers

In some cases the VN supplied to a customer may be built using resources from different technology layers operated by different operators. For example, one operator may run a packet TE network and use optical connectivity provided by another operator.

As shown in Figure 10, a customer asks for end-to-end connectivity between CE A and CE B, a virtual network. The customer's CNC makes a request to Operator 1's MDSC. The MDSC works out which network resources need to be configured and sends instructions to the appropriate PNCs. However, the link between Q and R is a virtual link supplied by Operator 2: Operator 1 is a customer of Operator 2.

To support this, Operator 1 has a CNC that communicates to Operator 2's MDSC. Note that Operator 1's CNC in Figure 10 is a functional component that does not dictate implementation: it may be embedded in a PNC.



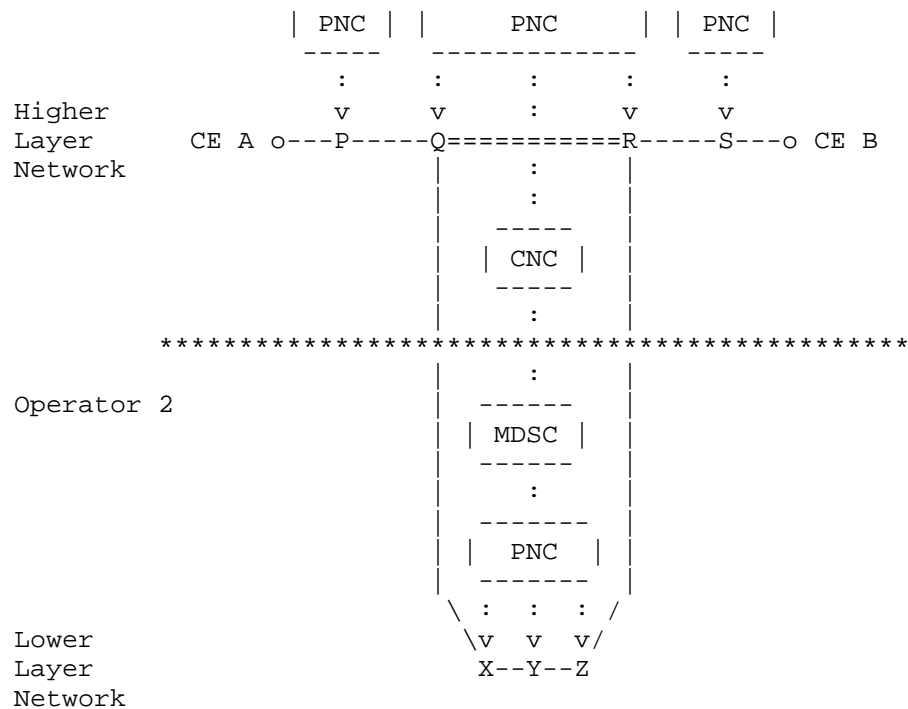


Figure 10: VN recursion with Network Layers

6. Access Points and Virtual Network Access Points

In order to map identification of connections between the customer's sites and the TE networks and to scope the connectivity requested in the VNS, the CNC and the MDSC refer to the connections using the Access Point (AP) construct as shown in Figure 11.



Figure 11: Customer View of APs

Let's take as an example a scenario shown in Figure 11. CE1 is connected to the network via a 10 Gbps link and CE2 via a 40 Gbps link. Before the creation of any VN between AP1 and AP2 the customer view can be summarized as shown in Table 1.

+-----+-----+			
End Point		Access Link Bandwidth	
+-----+-----+			
AP id	CE,port	MaxResBw	AvailableBw
+-----+-----+			
AP1	CE1,portX	10 Gbps	10 Gbps
+-----+-----+			
AP2	CE2,portZ	40 Gbps	40 Gbps
+-----+-----+			

Table 1: AP - Customer View

On the other hand, what the operator sees is shown in Figure 12.

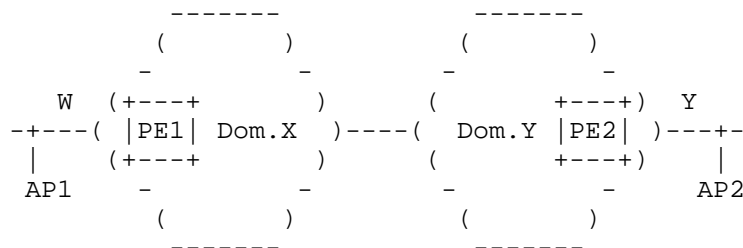


Figure 12: Operator view of the AP

Which results in a summarization as shown in Table 2.

End Point		Access Link Bandwidth	
AP id	PE,port	MaxResBw	AvailableBw
AP1	PE1,portW	10 Gbps	10 Gbps
AP2	PE2,portY	40 Gbps	40 Gbps

Table 2: AP - Operator View

A Virtual Network Access Point (VNAP) needs to be defined as binding between an AP and a VN. It is used to allow for different VNs to start from the same AP. It also allows for traffic engineering on the access and/or inter-domain links (e.g., keeping track of bandwidth allocation). A different VNAP is created on an AP for each VN.

In this simple scenario we suppose we want to create two virtual networks. The first with VN identifier 9 between AP1 and AP2 with bandwidth of 1 Gbps, while the second with VN identifier 5, again between AP1 and AP2 and with bandwidth 2 Gbps.

The operator view would evolve as shown in Table 3.

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

Table 3: AP and VNAP - Operator View after VNS Creation

6.1. Dual-Homing Scenario

Often there is a dual homing relationship between a CE and a pair of PEs. This case needs to be supported by the definition of VN, APs, and VNAPs. Suppose CE1 connected to two different PEs in the

operator domain via AP1 and AP2 and that the customer needs 5 Gbps of bandwidth between CE1 and CE2. This is shown in Figure 12.

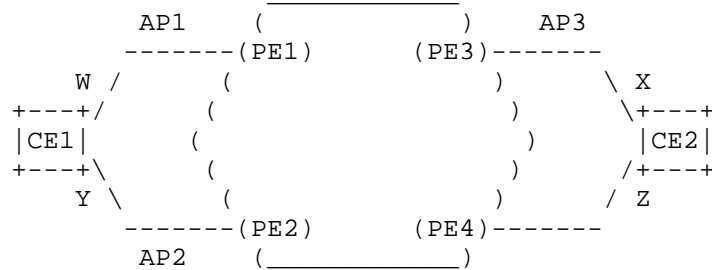


Figure 12: Dual-Homing Scenario

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

The customer view would be shown in Table 4.

	End Point	Access Link/VNAP Bw		
AP/VNAPid	CE,port	MaxResBw	AvailableBw	Dual Homing
AP1 -VNAP1.9	CE1,portW	10 Gbps 5 Gbps	5 Gbps N.A.	VNAP2.9
AP2 -VNAP2.9	CE1,portY	40 Gbps 5 Gbps	35 Gbps N.A.	VNAP1.9
AP3 -VNAP3.9	CE2,portX	50 Gbps 5 Gbps	45 Gbps N.A.	NONE

Table 4: Dual-Homing - Customer View after VN Creation

7. Advanced ACTN Application: Multi-Destination Service

A further advanced application of ACTN is in the case of Data Center selection, where the customer requires the Data Center selection to be based on the network status; this is referred to as Multi-Destination in [ACTN-REQ]. In terms of ACTN, a CNC could request a

VNS between a set of source APs and destination APs and leave it up to the network (MDSC) to decide which source and destination access points to be used to set up the VNS. The candidate list of source and destination APs is decided by a CNC (or an entity outside of ACTN) based on certain factors which are outside the scope of ACTN.

Based on the AP selection as determined and returned by the network (MDSC), the CNC (or an entity outside of ACTN) should further take care of any subsequent actions such as orchestration or service setup requirements. These further actions are outside the scope of ACTN.

Consider a case as shown in Figure 14, where three data centers are available, but the customer requires the data center selection to be based on the network status and the connectivity service setup between the AP1 (CE1) and one of the destination APs (AP2 (DC-A), AP3 (DC-B), and AP4 (DC-C)). The MDSC (in coordination with PNCs) would select the best destination AP based on the constraints, optimization criteria, policies, etc., and setup the connectivity service (virtual network).

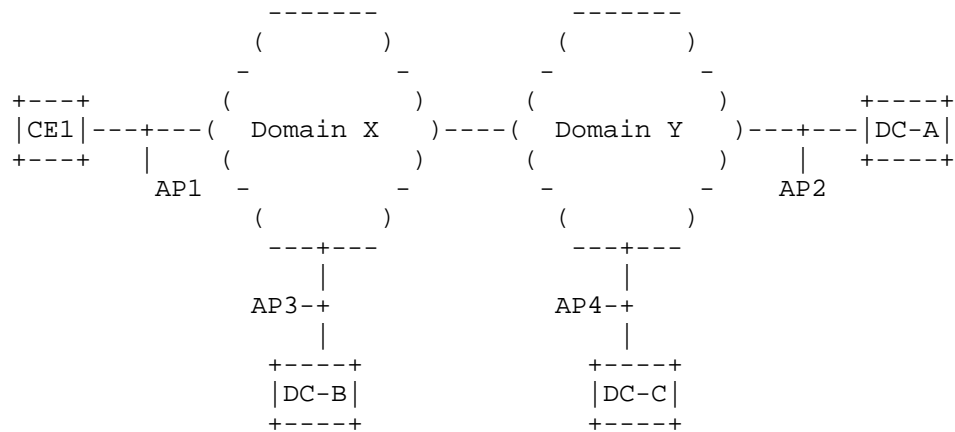


Figure 14: End-Point Selection Based on Network Status

7.1. Pre-Planned End Point Migration

Furthermore, in case of Data Center selection, customer could request for a backup DC to be selected, such that in case of failure, another DC site could provide hot stand-by protection. As shown in Figure 15 DC-C is selected as a backup for DC-A. Thus, the VN should be setup by the MDSC to include primary connectivity

between AP1 (CE1) and AP2 (DC-A) as well as protection connectivity between AP1 (CE1) and AP4 (DC-C).

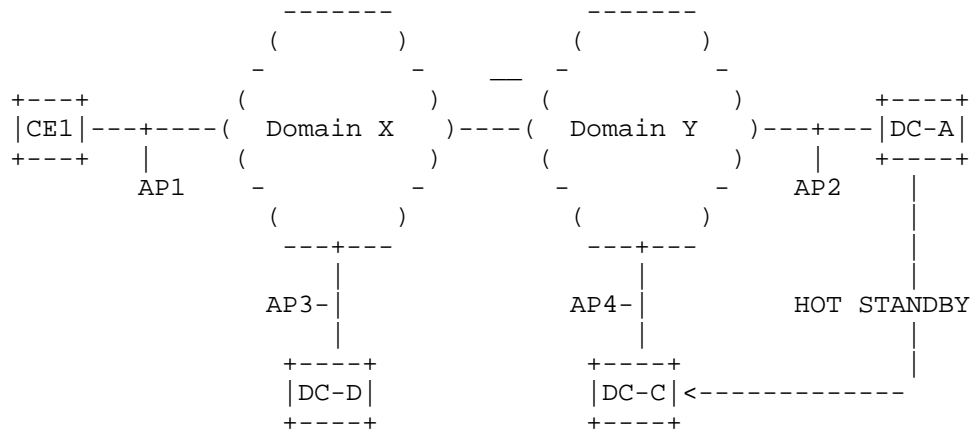


Figure 15: Pre-planned End-Point Migration

7.2. On the Fly End-Point Migration

Compared to pre-planned end point migration, on the fly end point selection is dynamic in that the migration is not pre-planned but decided based on network condition. Under this scenario, the MDSC would monitor the network (based on the VN Service-level Agreement (SLA) and notify the CNC in case where some other destination AP would be a better choice based on the network parameters. The CNC should instruct the MDSC when it is suitable to update the VN with the new AP if it is required.

8. Manageability Considerations

The objective of ACTN is to manage traffic engineered resources, and provide a set of mechanisms to allow customers to request virtual connectivity across server network resources. ACTN supports multiple customers each with its own view of and control of a virtual network built on the server network, the network operator will need to partition (or "slice") their network resources, and manage the resources accordingly.

The ACTN platform will, itself, need to support the request, response, and reservations of client and network layer connectivity. It will also need to provide performance monitoring and control of traffic engineered resources. The management requirements may be categorized as follows:

- . Management of external ACTN protocols
- . Management of internal ACTN interfaces/protocols
- . Management and monitoring of ACTN components
- . Configuration of policy to be applied across the ACTN system

The ACTN framework and interfaces are defined to enable traffic engineering for virtual network services and connectivity services. Network operators may have other Operations, Administration, and Maintenance (OAM) tasks for service fulfillment, optimization, and assurance beyond traffic engineering. The realization of OAM beyond abstraction and control of traffic engineered networks is not considered in this document.

8.1. Policy

Policy is an important aspect of ACTN control and management. Policies are used via the components and interfaces, during deployment of the service, to ensure that the service is compliant with agreed policy factors and variations (often described in SLAs), these include, but are not limited to: connectivity, bandwidth, geographical transit, technology selection, security, resilience, and economic cost.

Depending on the deployment of the ACTN architecture, some policies may have local or global significance. That is, certain policies may be ACTN component specific in scope, while others may have broader scope and interact with multiple ACTN components. Two examples are provided below:

- o A local policy might limit the number, type, size, and scheduling of virtual network services a customer may request via its CNC. This type of policy would be implemented locally on the MDSC.
- o A global policy might constrain certain customer types (or specific customer applications) to only use certain MDSCs, and be restricted to physical network types managed by the PNCs. A global policy agent would govern these types of policies.

The objective of this section is to discuss the applicability of ACTN policy: requirements, components, interfaces, and examples. This section provides an analysis and does not mandate a specific method for enforcing policy, or the type of policy agent that would

be responsible for propagating policies across the ACTN components. It does highlight examples of how policy may be applied in the context of ACTN, but it is expected further discussion in an applicability or solution specific document, will be required.

8.2. Policy Applied to the Customer Network Controller

A virtual network service for a customer application will be requested by the CNC. The request will reflect the application requirements and specific service needs, including bandwidth, traffic type and survivability. Furthermore, application access and type of virtual network service requested by the CNC, will be need adhere to specific access control policies.

8.3. Policy Applied to the Multi-Domain Service Coordinator

A key objective of the MDSC is to support the customer's expression of the application connectivity request via its CNC as a set of desired business needs, therefore policy will play an important role.

Once authorized, the virtual network service will be instantiated via the CNC-MDSC Interface (CMI); it will reflect the customer application and connectivity requirements, and specific service transport needs. The CNC and the MDSC components will have agreed connectivity end-points; use of these end-points should be defined as a policy expression when setting up or augmenting virtual network services. Ensuring that permissible end-points are defined for CNCs and applications will require the MDSC to maintain a registry of permissible connection points for CNCs and application types.

Conflicts may occur when virtual network service optimization criteria are in competition. For example, to meet objectives for service reachability a request may require an interconnection point between multiple physical networks; however, this might break a confidentiality policy requirement of specific type of end-to-end service. Thus an MDSC may have to balance a number of the constraints on a service request and between different requested services. It may also have to balance requested services with operational norms for the underlying physical networks. This balancing may be resolved using configured policy and using hard and soft policy constraints.

8.4. Policy Applied to the Provisioning Network Controller

The PNC is responsible for configuring the network elements, monitoring physical network resources, and exposing connectivity

(direct or abstracted) to the MDSC. It is therefore expected that policy will dictate what connectivity information will be exported between the PNC, via the MDSC-PNC Interface (MPI), and MDSC.

Policy interactions may arise when a PNC determines that it cannot compute a requested path from the MDSC, or notices that (per a locally configured policy) the network is low on resources (for example, the capacity on key links become exhausted). In either case, the PNC will be required to notify the MDSC, which may (again per policy) act to construct a virtual network service across another physical network topology.

Furthermore, additional forms of policy-based resource management will be required to provide virtual network service performance, security and resilience guarantees. This will likely be implemented via a local policy agent and additional protocol methods.

9. Security Considerations

The ACTN framework described in this document defines key components and interfaces for managed traffic engineered networks. Securing the request and control of resources, confidentiality of the information, and availability of function, should all be critical security considerations when deploying and operating ACTN platforms.

Several distributed ACTN functional components are required, and implementations should consider encrypting data that flows between components, especially when they are implemented at remote nodes, regardless these data flows are on external or internal network interfaces.

The ACTN security discussion is further split into two specific categories described in the following sub-sections:

- o Interface between the Customer Network Controller and Multi-Domain Service Coordinator (MDSC), CNC-MDSC Interface (CMI)
- o Interface between the Multi-Domain Service Coordinator and Provisioning Network Controller (PNC), MDSC-PNC Interface (MPI)

From a security and reliability perspective, ACTN may encounter many risks such as malicious attack and rogue elements attempting to connect to various ACTN components. Furthermore, some ACTN components represent a single point of failure and threat vector, and must also manage policy conflicts, and eavesdropping of communication between different ACTN components.

The conclusion is that all protocols used to realize the ACTN framework should have rich security features, and customer, application and network data should be stored in encrypted data stores. Additional security risks may still exist. Therefore, discussion and applicability of specific security functions and protocols will be better described in documents that are use case and environment specific.

9.1. CNC-MDSC Interface (CMI)

Data stored by the MDSC will reveal details of the virtual network services, and which CNC and customer/application is consuming the resource. The data stored must therefore be considered as a candidate for encryption.

CNC Access rights to an MDSC must be managed. The MDSC must allocate resources properly, and methods to prevent policy conflicts, resource wastage, and denial of service attacks on the MDSC by rogue CNCs, should also be considered.

The CMI will likely be an external protocol interface. Suitable authentication and authorization of each CNC connecting to the MDSC will be required, especially, as these are likely to be implemented by different organizations and on separate functional nodes. Use of the AAA-based mechanisms would also provide role-based authorization methods, so that only authorized CNC's may access the different functions of the MDSC.

9.2. MDSC-PNC Interface (MPI)

Where the MDSC must interact with multiple (distributed) PNCs, a PKI-based mechanism is suggested, such as building a TLS or HTTPS connection between the MDSC and PNCs, to ensure trust between the physical network layer control components and the MDSC. Trust anchors for the PKI can be configured to use a smaller (and potentially non-intersecting) set of trusted Certificate Authorities (CAs) than in the Web PKI.

Which MDSC the PNC exports topology information to, and the level of detail (full or abstracted), should also be authenticated, and specific access restrictions and topology views should be configurable and/or policy-based.

10. IANA Considerations

This document has no actions for IANA.

11. References

11.1. Informative References

- [RFC2702] Awduche, D., et. al., "Requirements for Traffic Engineering Over MPLS", RFC 2702, September 1999.
- [RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", IETF RFC 4655, August 2006.
- [RFC5654] Niven-Jenkins, B. (Ed.), D. Brungard (Ed.), and M. Betts (Ed.), "Requirements of an MPLS Transport Profile", RFC 5654, September 2009.
- [RFC7149] Boucadair, M. and Jacquenet, C., "Software-Defined Networking: A Perspective from within a Service Provider Environment", RFC 7149, March 2014.
- [RFC7926] A. Farrel (Ed.), "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.
- [RFC3945] Manning, E., et al., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture2", RFC 3945, October 2004.
- [ONF-ARCH] Open Networking Foundation, "SDN architecture", Issue 1.1, ONF TR-521, June 2016.
- [Centralized] Farrel, A., et al., "An Architecture for Use of PCE and PCEP in a Network with Central Control", draft-ietf-teas-pce-central-control, work in progress.
- [Service-YANG] Lee, Y., Dhody, D., and Ceccarelli, C., "Traffic Engineering and Service Mapping Yang Model", draft-lee-teas-te-service-mapping-yang, work in progress.
- [ACTN-YANG] Lee, Y., et al., "A Yang Data Model for ACTN VN Operation", draft-lee-teas-actn-vn-yang, work in progress.
- [ACTN-REQ] Lee, Y., et al., "Requirements for Abstraction and Control of TE Networks", draft-ietf-teas-actn-requirements, work in progress.
- [TE-Topo] X. Liu et al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress.

12. Contributors

Adrian Farrel
Old Dog Consulting
Email: adrian@olddog.co.uk

Italo Busi
Huawei
Email: Italo.Busi@huawei.com

Khuzema Pithewan
Infinera
Email: kpithewan@infinera.com

Michael Scharf
Nokia
Email: michael.scharf@nokia.com

Luyuan Fang
eBay
Email: luyuanf@gmail.com

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

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

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

Dhruv Dhody
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
Email: dhruv.ietf@gmail.com

Gert Grammel
 Juniper Networks
 Email: ggrammel@juniper.net

Authors' Addresses

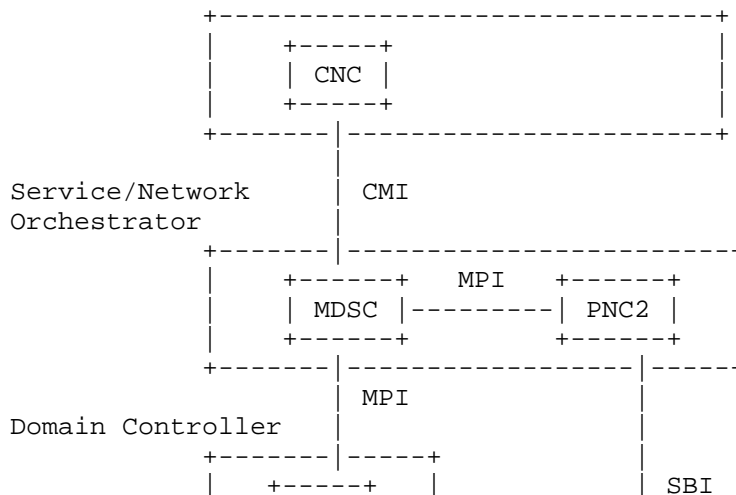
Daniele Ceccarelli
 Ericsson
 Torshamnsgatan, 48
 Stockholm, Sweden
 Email: daniele.ceccarelli@ericsson.com

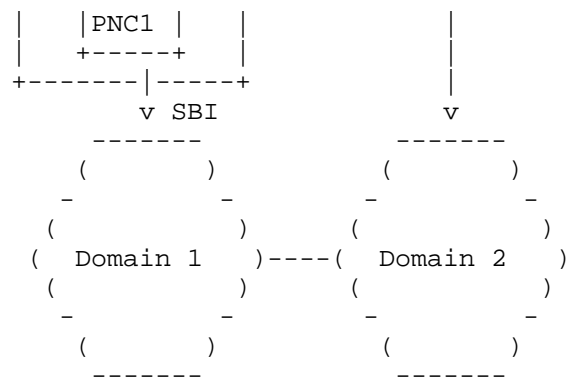
Young Lee
 Huawei Technologies
 5340 Legacy Drive
 Plano, TX 75023, USA
 Phone: (469)277-5838
 Email: leeyoung@huawei.com

APPENDIX A - Example of MDSC and PNC Functions Integrated in A Service/Network Orchestrator

This section provides an example of a possible deployment scenario, in which Service/Network Orchestrator can include a number of functionalities, among which, in the example below, PNC functionalities for domain 2 and MDSC functionalities to coordinate the PNC1 functionalities (hosted in a separate domain controller) and PNC2 functionalities (co-hosted in the network orchestrator).

Customer





TEAS Working Group
Internet Draft
Intended status: Informational

Expires September 1, 2018.

Young Lee (Editor)
Huawei

Daniele Ceccarelli
Ericsson

Takuya Miyasaka
KDDI

Jong Yoon Shin
SKT

Kwang-koog Lee
KT

March 1, 2018

Requirements for Abstraction and Control of TE Networks

draft-ietf-teas-actn-requirements-09

Abstract

This document provides a set of functional requirements for abstraction and control of Traffic Engineering networks to facilitate virtual network operation via the creation of a single virtualized network or a seamless service. This supports operators in viewing and controlling different domains (at any dimension: applied technology, administrative zones, or vendor-specific technology islands) as a single virtualized network.

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 September 1, 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. Requirements Language.....	4
2. High-level ACTN requirements.....	4
2.1. Service-Specific Requirements.....	5
2.2. Network-Related Requirements.....	7
3. Security Considerations.....	9
4. IANA Considerations.....	9
5. References.....	10
5.1. Normative References.....	10
5.2. Informative References.....	10
6. Contributors.....	11
Authors' Addresses.....	12

1. Introduction

This document provides a set of functional requirements for Abstraction and Control of Traffic Engineering (TE) Networks (ACTN) identified in various use-cases specified by the operators. [ACTN-Frame] defines the base reference architecture and terminology.

ACTN refers to the set of virtual network service operations needed to coordinate, control and manage large-scale multi-domain TE networks so as to facilitate network programmability, automation, efficient resource sharing, and end-to-end virtual service aware connectivity.

These operations are summarized as follows:

- Abstraction and coordination of underlying network resources independent of how these resources are managed or controlled, so that higher-layer entities can dynamically control virtual networks based on those resources. Control includes creating, modifying, monitoring, and deleting virtual networks.
- Collation of the identifiers and other attributes of the resources from multiple TE networks (multiple technologies, equipment from multiple vendors, under the control of multiple administrations) through a process of recursive abstraction to present a customer with a single virtual network. This is achieved by presenting the network domain as an abstracted topology to the customer via open and programmable interfaces. Recursive abstraction allows for the recursion of abstracted data in a hierarchy of controllers.. It is expected that the recursion levels should be at least three levels: customer level, multi-domain network level, and domain network level.
- Coordination of end-to-end virtual network services and applications via allocation of network resources to meet specific service, application and customer requirements. Refer to [ACTN-Frame] for the definition of coordination.
- Adaptation of customer requests (to control virtual resources) to the physical network resources performing the necessary mapping, translation, isolation and, policy that allows conveying, managing and enforcing customer policies with respect to the services and the network of the customer.

- Provision via a data model and virtual control capability to customers who request virtual network services. Note that these customers could, themselves, be service providers.

ACTN solutions will build on, and extend, existing TE constructs and TE mechanisms wherever possible and appropriate. Support for controller-based approaches is specifically included in the possible solution set.

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. High-level ACTN requirements

This section provides a summary of use-cases in terms of two categories: (i) service-specific requirements; (ii) network-related requirements. All these requirements are specified by operators that are interested in implementing ACTN.

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

Network-related requirements are related to and necessary for coherent/seamless for the virtual network operation function. These requirements are related to multi-domain and multi-layer signaling, routing, protection/restoration and re-optimization/re-grooming, etc.

Each requirement specified in Sections 2.1 and 2.2 is derived from ACTN use-cases: [CHENG], [DHODY], [FANG], [KLEE], [KUMAKI], [LOPEZ], [SHIN], [XU], [XU2], and [SUZUKI].

2.1. Service-Specific Requirements

1. Requirement 1: Virtual Network Service (VNS) creation

Customer MUST be able to request/instantiate the VNS to the network within the confines of mutual agreement between customer and network operator and network operator's capability. A VNS is the service agreement between a customer and provider to provide a VN [ACTN-Frame]. There are different types of VNS in terms of the VN types the customer is allowed to operate (e.g., a VN type can be simply a set of edge-to-edge links, or it can comprise of virtual nodes and virtual links, etc.). The customer MUST be able to express VNS preference that captures Service Level Agreements (SLA) associated with virtual network service (e.g., Endpoint selection preference, routing preference, time-related preference, etc.)

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

2. Requirement 2: Virtual Network Service Query

Customer SHOULD be able to request VNS Query ("Can you give me these VN(s)?") that include the following parameters:

- VN type: various VN types defined by the customer (e.g., path, graph, etc.)
- VN end-points (Customer Edge interface information)
- VN Topology Service-specific Objective Functions (e.g., a set of objective functions as defined in [RFC5541] to be supported on the paths, but not limited to).
- VN constraints requirement (e.g., Maximum Latency threshold, Minimum Bandwidth, etc.)

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

3. Requirement 3: VNS Instantiation ("Please create a VNS for me")

Customer MUST be able to instantiate VNS that includes various VNS related parameters:

- VN type: various VN types defined by the customer (e.g., Type 1, Type 2, etc. See [ACTN-Frame] for the definition of VN Type 1 and Type 2).
- VN end-points (Customer Edge interface information)
- VN Topology Service-specific Objective Functions (e.g., a set of objective functions as defined in [RFC5541] to be supported on the paths, but not limited to).
- VN constraints requirement (e.g., Maximum Latency threshold, Minimum Bandwidth, etc.)
- VN Topology diversity when there are multiple instances of VNS (e.g., VN1 and VN2 must be disjoint; Node/link disjoint from other VNs)

Note that Requirement 3 provides specific details of Requirement 1.

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

4. Requirement 4: VNS Lifecycle Management & Operation (M&O)

Customer MUST be able to perform the following VNS operations:

- VNS Delete: Customer MUST be able to delete VNS.
- VNS Modify: Customer MUST be able to modify VNS related parameters during the lifecycle of the instantiated VNS.

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

5. Requirement 5: VNS Isolation

Customer's VN should be able to use arbitrary network topology, routing, or forwarding functions as well as customized control mechanisms independent of the underlying physical network and of other coexisting virtual networks. Other customers' VNS operation MUST NOT impact a particular customer's VNS network operation.

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

6. Requirement 6: Multi-Destination Coordination

Customer MUST be able to define and convey service/preference requirements for multi-destination applications (e.g., set of candidate sources/destinations, thresholds for load balancing, disaster recovery preference, etc.)

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

7. Requirement 7: VNS Performance Monitoring

The customer MUST be able to define performance monitoring parameters and its associated preference such as frequency of report, abstraction/aggregation level of performance data (e.g., VN level, tunnel level, etc.) with dynamic feedback loop from the network.

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

8. Requirement 8: VNS Confidentiality and Security Requirements

The following confidentiality/security requirements MUST be supported in all interfaces:

- Securing the request and control of resources, confidentiality of the information, and availability of function.
- Trust domain verification between a customer entity and a network entity. It verifies if a trust relationship has been established between these entities.
- Encrypting data that flow between components, especially when they are implemented at remote nodes, regardless if these are external or internal network interfaces.

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

2.2. Network-Related Requirements

1. Requirement 1: Virtual Network Service Coordination

Network MUST be able to support the following VNS operations:

- VNS Create: Upon customer's VNS creation request, network MUST be able to create VNS within the confines of network resource availability.
- VNS Delete: Upon customer's VNS deletion request, network MUST be able to delete VNS.
- VNS Modify: Upon customer's VNS modification request, network MUST be able to modify VNS related parameters during the lifecycle of the instantiated VNS.
- VNS Monitor: Upon customer's VNS performance monitoring setup, the network MUST be able to support VNS level Operations, Administration and Management (OAM) Monitoring under service agreement.

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

2. Requirement 2: Topology Abstraction Capability

The network MUST be capable of managing its networks based on the principle of topology abstraction to be able to scale multi-layer, multi-domain networks.

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

3. Requirement 3: Multi-Domain & Multi-layer Coordination

Network coordination for multi-domain and multi-layer path computation and path setup operation MUST be provided:

- End-to-end path computation across multi-domain networks (based on abstract topology from each domain)
- Domain sequence determination
- Request for path signaling to each domain controller
- Alternative TE path computation if any of the domain controllers cannot find its domain path

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

4. Requirement 4: End-to-End Path Protection

End-to-end Path Protection Operations MUST be provided with seamless coordination between domain-level protection schemes and cross-domain protection schemes.

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

5. Requirement 5: Dynamicity of virtual network control operations

Dynamic virtual network control operations MUST be supported. This includes, but is not limited to, the following:

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

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

3. Security Considerations

The ACTN requirements described in this document do not directly bear specific security concerns. When these requirements are implemented in specific interfaces, securing the request and control of resources, confidentiality of the information, and availability of function, should all be critical security considerations.

4. IANA Considerations

This document has no actions for IANA.

5. References

5.1. Normative References

- [ACTN-Frame] D. Ceccarelli, et al., "Framework for Abstraction and Control of Transport Networks", draft-ietf-teas-actn-framework, work in progress.

5.2. Informative References

- [RFC2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.
- [RFC5541] JL. Le Roux, JP. Vasseur, and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, June 2009.
- [RFC8174] B. Leiba, "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [CHENG] W. Cheng, et. al., "ACTN Use-cases for Packet Transport Networks in Mobile Backhaul Networks", draft-cheng-actn-ptn-requirements-00, July 21, 2014.
- [DHODY] D. Dhody, et. al., "Packet Optical Integration (POI) Use Cases for Abstraction and Control of Transport Networks (ACTN)", draft-dhody-actn-poi-use-case-07, October 28, 2016.
- [FANG] L. Fang, "ACTN Use Case for Multi-domain Data Center Interconnect", draft-fang-actn-multidomain-dci-01, September 29, 2014.
- [KLEE] K. Lee, H. Lee, R. Vilata, V. Lopez, "ACTN Use-case for E2E Network Services in Multiple Vendor Domain Transport Networks", draft-klee-teas-actn-connectivity-multi-domain-03, July 31, 2017.
- [KUMAKI] K. Kumaki, T. Miyasaka, "ACTN : Use case for Multi Tenant VNO", draft-kumaki-teas-actn-multitenant-vno-00, May 29, 2014.
- [LOPEZ] D. Lopez (Ed), "ACTN Use-case for Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-lopez-actn-vno-multidomains-01, October 27, 2014.

- [SHIN] J. Shin, R. Hwang, J. Lee, "ACTN Use-case for Mobile Virtual Network Operation for Multiple Domains in a Single Operator Network", draft-shin-actn-mvno-multi-domain-00, June 30, 2014.
- [XU] Y. Xu, et. al., "Use Cases and Requirements of Dynamic Service Control based on Performance Monitoring in ACTN Architecture", draft-xu-actn-perf-dynamic-service-control-03, April 23, 2015.
- [XU2] Y. Xu, et. al., "Requirements of Abstract Alarm Report in ACTN architecture", draft-xu-teas-actn-abstract-alarm-report-00, July 6, 2015.
- [SUZUKI] T. Suzuki, et. al., "Use-case and Requirements for Multi-domain Operation Plane Change", draft-suzuki-teas-actn-multidomain-opc-00, July 6, 2015.

6. Contributors

Dhruv Dhody
Huawei Technologies
Email: dhruv.ietf@gmail.com

Sergio Belotti
Nokia
Via Trento, 30
Vimercate, Italy
Email: sergio.belotti@nokia.com

Khuzema Pithewan
Peloton Technology
Email: khuzemap@gmail.com

Yunbin Xu
CATR
Email: xuyunbin@ritt.cn

Toshiaki Suzuki
Hitachi

Email: toshiaki.suzuki.cs@hitachi.com

Haomian Zheng
Huawei
Email: zhenghaomian@huawei.com

Authors' Addresses

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

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

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

Jong Yoon Shin
SKT
Email: jongyoon.shin@sk.com

Kwang-koog Lee
KT
Email: kwangkoog.lee@kt.com

TEAS Working Group
Internet-Draft
Updates: 4090
Intended Status: Standards Track
Expires: March 1, 2018

M. Taillon
T. Saad, Ed.
R. Gandhi, Ed.
Z. Ali
Cisco Systems, Inc.
M. Bhatia
Nokia
August 28, 2017

Updates to Resource Reservation Protocol For Fast Reroute of
Traffic Engineering GMPLS LSPs
draft-ietf-teas-gmpls-lsp-fastreroute-12

Abstract

This document updates the Resource Reservation Protocol - Traffic Engineering (RSVP-TE) Fast Reroute (FRR) procedures defined in RFC 4090 to support Packet Switched Capable (PSC) Generalized Multi-Protocol Label Switching (GMPLS) Label Switched Paths (LSPs). These updates allow the coordination of a bidirectional bypass tunnel assignment protecting a common facility in both forward and reverse directions of a co-routed bidirectional LSP. In addition, these updates enable the re-direction of bidirectional traffic onto bypass tunnels that ensure co-routedness of data paths in the forward and reverse directions after FRR and avoid RSVP soft-state timeout in control-plane.

Status of this Memo

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

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

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

Copyright Notice

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

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

Table of Contents

1. Introduction	4
2. Conventions Used in This Document	5
2.1. Key Word Definitions	5
2.2. Terminology	5
2.3. Abbreviations	6
3. Fast Reroute For Unidirectional GMPLS LSPs	6
4. Bypass Tunnel Assignment For Bidirectional GMPLS LSPs	6
4.1. Bidirectional GMPLS Bypass Tunnel Direction	7
4.2. Merge Point Labels	7
4.3. Merge Point Addresses	7
4.4. RRO IPv4/IPv6 Subobject Flags	8
4.5. Bidirectional Bypass Tunnel Assignment Co-ordination	8
4.5.1. Bidirectional Bypass Tunnel Assignment Signaling Procedure	8
4.5.2. One-to-one Bidirectional Bypass Tunnel Assignment	10
4.5.3. Multiple Bidirectional Bypass Tunnel Assignments	10
5. Fast Reroute For Bidirectional GMPLS LSPs with In-band Signaling	11
5.1. Link Protection for Bidirectional GMPLS LSPs	12
5.1.1. Behavior After Link Failure	12
5.1.2. Revertive Behavior After Fast Reroute	12
5.2. Node Protection for Bidirectional GMPLS LSPs	13
5.2.1. Behavior After Link Failure	14
5.2.2. Behavior After Link Failure To Re-coroute	14
5.2.2.1. Re-coroute in Data-plane After Link Failure	15
5.2.3. Revertive Behavior After Fast Reroute	15
5.2.4. Behaviour After Node Failure	16
5.3. Unidirectional Link Failures	16
6. Fast Reroute For Bidirectional GMPLS LSPs with Out-of-band Signaling	17

7. Message and Object Definitions	17
7.1. BYPASS_ASSIGNMENT Subobject	17
7.2. FRR Bypass Assignment Error Notify Message	19
8. Compatibility	19
9. Security Considerations	19
10. IANA Considerations	20
10.1. BYPASS_ASSIGNMENT Subobject	20
10.2. FRR Bypass Assignment Error Notify Message	20
11. References	22
11.1. Normative References	22
11.2. Informative References	22
Acknowledgements	23
Contributors	23
Authors' Addresses	24

1. Introduction

Packet Switched Capable (PSC) Traffic Engineering (TE) Label Switched Paths (LSPs) can be setup using Generalized Multi-Protocol Label Switching (GMPLS) signaling procedures specified in [RFC3473] for both unidirectional and bidirectional tunnels. The GMPLS signaling allows sending and receiving the RSVP messages in-band with the data traffic or out-of-band over a separate control-channel. Fast Reroute (FRR) [RFC4090] has been widely deployed in the packet TE networks today and is desirable for TE GMPLS LSPs. Using FRR methods also allows the leveraging of the existing mechanisms for failure detection and restoration in deployed networks.

The FRR procedures defined in [RFC4090] describe the behavior of the Point of Local Repair (PLR) to reroute traffic and signaling onto the bypass tunnel in the event of a failure for protected LSPs. Those procedures are applicable to the unidirectional protected LSPs signaled using either RSVP-TE [RFC3209] or GMPLS procedures [RFC3473]. When using the FRR procedures defined in [RFC4090] with co-routed bidirectional GMPLS LSPs, it is desired that same PLR and Merge Point (MP) pairs are selected in each direction and both PLR and MP assign the same bidirectional bypass tunnel. This document updates the FRR procedures defined in [RFC4090] to coordinate the bidirectional bypass tunnel assignment and to exchange MP labels between upstream and downstream PLRs of the protected co-routed bidirectional LSP.

When using FRR procedures with co-routed bidirectional GMPLS LSPs, it is possible in some cases for the RSVP signaling refreshes to stop reaching certain nodes along the protected LSP path after the PLRs finish rerouting of the signaling messages. This can occur after a failure event when using node protection bypass tunnels. As shown in Figure 2, this is possible even with selecting the same bidirectional bypass tunnels in both directions and the same PLR and MP pairs. This is caused by the asymmetry of paths that may be taken by the bidirectional LSP's signaling in the forward and reverse directions due to upstream and downstream PLRs independently triggering FRR. In such cases, after FRR, the RSVP soft-state timeout causes the protected bidirectional LSP to be torn down, with subsequent traffic loss.

Protection State Coordination Protocol [RFC6378] is applicable to FRR [RFC4090] for local protection of co-routed bidirectional LSPs in order to minimize traffic disruptions in both directions. However, this does not address the above mentioned problem of RSVP soft-state timeout that can occur in the control-plane.

This document defines a solution to the RSVP soft-state timeout issue

by providing mechanisms in the control-plane to complement the FRR procedures of [RFC4090]. The solution allows to maintain the RSVP soft-state for co-routed bidirectional protected GMPLS LSPs in the control-plane and achieve co-routedness of the paths followed by the traffic in the forward and reverse directions after FRR.

The procedures defined in this document apply to GMPLS signaled PSC TE co-routed bidirectional protected LSPs and co-routed bidirectional FRR bypass tunnels. Unless otherwise specified in this document, the FRR procedures defined in [RFC4090] are not modified by this document. The FRR mechanism for associated bidirectional GMPLS LSPs where two unidirectional GMPLS LSPs are bound together by using the association signaling [RFC7551] is outside the scope of this document.

2. Conventions Used in This Document

2.1. Key Word Definitions

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

2.2. Terminology

The reader is assumed to be familiar with the terminology in [RFC2205], [RFC3209], [RFC3471], [RFC3473], and [RFC4090].

Downstream PLR: Downstream Point of Local Repair. The PLR that locally detects a failure in the downstream direction of the traffic flow and reroutes traffic in the same direction of the protected bidirectional LSP RSVP Path signaling. A downstream PLR has a corresponding downstream MP.

Downstream MP: Downstream Merge Point. The LSR where one or more backup tunnels rejoin the path of the protected LSP in the downstream direction of the traffic flow. The same LSR can be both a downstream MP and an upstream PLR simultaneously.

Upstream PLR: Upstream Point of Local Repair. The PLR that locally detects a failure in the upstream direction of the traffic flow and reroutes traffic in the opposite direction of the protected bidirectional LSP RSVP Path signaling. An upstream PLR has a corresponding upstream MP.

Upstream MP: Upstream Merge Point. The LSR where one or more backup tunnels rejoin the path of the protected LSP in the upstream

direction of the traffic flow. The same LSR can be both an upstream MP and a downstream PLR simultaneously.

Point of Remote Repair (PRR): A downstream MP that assumes the role of upstream PLR upon receiving protected LSP's rerouted Path message and triggers reroute of traffic and signaling in the upstream direction of the traffic flow using the procedures described in this document.

2.3. Abbreviations

GMPLS: Generalized Multi-Protocol Label Switching

LSP: Label Switched Path

LSR: Label Switching Router

MP: Merge Point

MPLS: Multi-Protocol Label Switching

PLR: Point of Local Repair

PSC: Packet Switched Capable

RSVP: Resource ReSerVation Protocol

TE: Traffic Engineering

3. Fast Reroute For Unidirectional GMPLS LSPs

The FRR procedures defined in [RFC4090] for RSVP-TE signaling [RFC3209] are equally applicable to the unidirectional protected LSPs signaled using GMPLS [RFC3473] and are not modified by the updates defined in this document except the following.

When using the GMPLS out-of-band signaling [RFC3473], after a link failure event, the RSVP messages are not rerouted over the bypass tunnel by the downstream PLR but instead rerouted over a control-channel to the downstream MP.

4. Bypass Tunnel Assignment For Bidirectional GMPLS LSPs

This section describes signaling procedures for FRR bidirectional bypass tunnel assignment for GMPLS signaled PSC co-routed bidirectional TE LSPs for both in-band and out-of-band signaling.

4.1. Bidirectional GMPLS Bypass Tunnel Direction

This document defines procedures where bidirectional GMPLS bypass tunnels are signaled in the same direction as the protected GMPLS LSPs. In other words, the bidirectional GMPLS bypass tunnels originate on the downstream PLRs and terminate on the corresponding downstream MPs. As the originating downstream PLR has the policy information about the locally provisioned bypass tunnels, it always initiates the bypass tunnel assignment. The bidirectional GMPLS bypass tunnels originating from the upstream PLRs and terminating on the corresponding upstream MPs are outside the scope of this document.

4.2. Merge Point Labels

To correctly reroute data traffic over a node protection bypass tunnel, the downstream and upstream PLRs have to know, in advance, the downstream and upstream MP labels of the protected LSP so that data in the forward and reverse directions can be redirected through the bypass tunnel after FRR respectively.

[RFC4090] defines procedures for the downstream PLR to obtain the protected LSP's downstream MP label from recorded labels in the RECORD_ROUTE Object (RRO) of the RSVP Resv message received at the downstream PLR.

To obtain the upstream MP label, the procedures specified in [RFC4090] are used to record the upstream MP label in the RRO of the RSVP Path message of the protected LSP. The upstream PLR obtains the upstream MP label from the recorded labels in the RRO of the received RSVP Path message.

4.3. Merge Point Addresses

To correctly assign a bidirectional bypass tunnel, the downstream and upstream PLRs have to know, in advance, the downstream and upstream MP addresses.

[RFC4561] defines procedures for the downstream PLR to obtain the protected LSP's downstream MP address from the recorded Node-IDs in the RRO of the RSVP Resv message received at the downstream PLR.

To obtain the upstream MP address, the procedures specified in [RFC4561] are used to record upstream MP Node-ID in the RRO of the RSVP Path message of the protected LSP. The upstream PLR obtains the upstream MP address from the recorded Node-IDs in the RRO of the received RSVP Path message.

4.4. RRO IPv4/IPv6 Subobject Flags

RRO IPv4/IPv6 subobject flags are defined in [RFC4090], Section 4.4 and are equally applicable to the FRR procedure for the protected bidirectional GMPLS LSPs.

The procedures defined in [RFC4090] are used by the downstream PLR to signal the IPv4/IPv6 subobject flags upstream in the RRO of the RSVP Resv message of the protected LSP. Similarly, those procedures are used by the downstream PLR to signal the IPv4/IPv6 subobject flags downstream in the RRO of the RSVP Path message of the protected LSP.

4.5. Bidirectional Bypass Tunnel Assignment Co-ordination

This document defines signaling procedures and a new `BYPASS_ASSIGNMENT` subobject in the RSVP `RECORD_ROUTE` Object (RRO) used to co-ordinate the bidirectional bypass tunnel assignment between the downstream and upstream PLRs.

4.5.1. Bidirectional Bypass Tunnel Assignment Signaling Procedure

It is desirable to coordinate the bidirectional bypass tunnel selected at the downstream and upstream PLRs so that the rerouted traffic flows on co-routed paths after FRR. To achieve this, a new RSVP subobject is defined for RRO that identifies a bidirectional bypass tunnel that is assigned at a downstream PLR to protect a bidirectional LSP.

When the procedures defined in this document are in use, the `BYPASS_ASSIGNMENT` subobject MUST be added by each downstream PLR in the RSVP Path RRO message of the GMPLS signaled bidirectional protected LSP to record the downstream bidirectional bypass tunnel assignment. This subobject is sent in the RSVP Path RRO message every time the downstream PLR assigns or updates the bypass tunnel assignment. The downstream PLR can assign a bypass tunnel when processing the first Path message of the protected LSP as long as it has a topological view of the downstream MP and the traversed path information in ERO. For the protected LSP where the downstream MP cannot be determined from the first Path message (e.g. when using loose hops in ERO), the downstream PLR needs to wait for Resv message with RRO in order to assign a bypass tunnel. However, in both cases, the downstream PLR cannot update the data-plane until it receives Resv messages containing the MP labels.

The upstream PLR (downstream MP) simply reflects the bypass tunnel assignment in the reverse direction. The absence of `BYPASS_ASSIGNMENT` subobject in Path RRO means that the relevant node or interface is not protected by a bidirectional bypass tunnel.

Hence, the upstream PLR need not assign a bypass tunnel in the reverse direction.

When the BYPASS_ASSIGNMENT subobject is added in the Path RRO:

- o The IPv4 or IPv6 subobject containing Node-ID address MUST also be added [RFC4561]. The Node-ID address MUST match the source address of the bypass tunnel selected for this protected LSP.
- o The BYPASS_ASSIGNMENT subobject MUST be added immediately after the Node-ID address.
- o The Label subobject MUST also be added [RFC3209].

The rules for adding an IPv4 or IPv6 Interface address subobject and Unnumbered Interface ID subobject as specified in [RFC3209] and [RFC4090] are not modified by the above procedure. The options specified in Section 6.1.3 in [RFC4990] are also applicable as long as above mentioned rules are followed when using the FRR procedures defined in this document.

An upstream PLR (downstream MP) SHOULD check all BYPASS_ASSIGNMENT subobjects in the Path RRO to see if the destination address in the BYPASS_ASSIGNMENT matches the address of the upstream PLR. For each BYPASS_ASSIGNMENT subobject that matches, the upstream PLR looks for a tunnel that has a source address matching the downstream PLR that inserted the BYPASS_ASSIGNMENT, as indicated by the Node-ID address, and the same tunnel-ID as indicated in the BYPASS_ASSIGNMENT. The RRO can contain multiple addresses to identify a node, however, the upstream PLR relies on the Node-ID address preceding the BYPASS_ASSIGNMENT subobject for identifying the bypass tunnel. If the bypass tunnel is not found, the upstream PLR SHOULD send a Notify message [RFC3473] with Error-code - FRR Bypass Assignment Error (value: TBA1) and Sub-code - Bypass Tunnel Not Found (value: TBA3) to the downstream PLR. Upon receiving this error, the downstream PLR SHOULD remove the bypass tunnel assignment and select an alternate bypass tunnel if one available. The RRO containing BYPASS_ASSIGNMENT subobject(s) is then simply forwarded downstream in the RSVP Path message.

A downstream PLR may add, remove or change bypass tunnel assignment for a protected LSP resulting in addition, removal or modification of BYPASS_ASSIGNMENT subobject in the Path RRO, respectively. In this case, the downstream PLR SHOULD generate modified Path message and forward it downstream. The downstream MP SHOULD check the RRO in the received Path message and update the bypass tunnel assignment in the reverse direction accordingly.

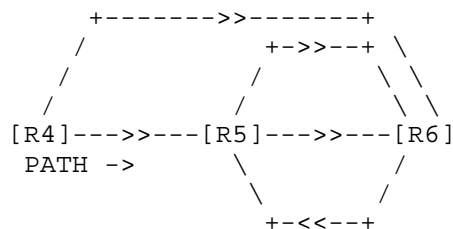
4.5.2. One-to-one Bidirectional Bypass Tunnel Assignment

The bidirectional bypass tunnel assignment co-ordination procedure defined in this document can be used for both facility backup described in Section 3.2 of [RFC4090] and one-to-one backup described in Section 3.1 of [RFC4090]. As specified in [RFC4090], Section 4.2, the DETOUR_OBJECT can be used in one-to-one backup method to identify the detour LSPs. In one-to-one backup method, if the bypass tunnel is already in-use at the upstream PLR, it SHOULD send a Notify message [RFC3473] with Error-code - FRR Bypass Assignment Error (value: TBA1) and Sub-code - One-to-one Bypass Already In-use (value: TBA4) to the downstream PLR. Upon receiving this error, the downstream PLR SHOULD remove the bypass tunnel assignment and select an alternate bypass tunnel if one available.

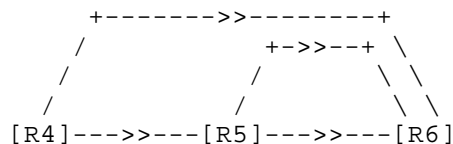
4.5.3. Multiple Bidirectional Bypass Tunnel Assignments

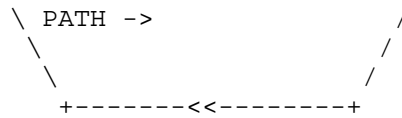
The upstream PLR may receive multiple bypass tunnel assignments for a protected LSP from different downstream PLRs leading to an asymmetric bypass tunnel assignment as shown in the following two examples.

As shown in Example 1 and Example 2, for the protected bidirectional GMPLS LSP R4-R5-R6, the upstream PLR R6 receives multiple bypass tunnel assignments, one from downstream PLR R4 for node protection and one from downstream PLR R5 for link protection. In Example 1, R6 prefers the link protection bypass tunnel from downstream PLR R5 whereas in Example 2, R6 prefers the node protection bypass tunnel from downstream PLR R4.



Example 1: Link protection is preferred on downstream MP





Example 2: Node protection is preferred on downstream MP

The asymmetry of bypass tunnel assignments can be avoided by using the flags in the SESSION_ATTRIBUTES Object defined in Section 4.3 of [RFC4090]. In particular, the "node protection desired" flag is signaled by the head-end node to request node protection bypass tunnels. When this flag is set, both downstream PLR and upstream PLR nodes assign node protection bypass tunnels as shown in Example 2. In the absence of "node protection desired" flag set, the downstream PLR nodes may only signal the link protection bypass tunnels avoiding the asymmetry of bypass tunnel assignments shown in Example 1.

When multiple bypass tunnel assignments are received, the upstream PLR SHOULD send a Notify message [RFC3473] with Error-code - FRR Bypass Assignment Error (value: TBA1) and Sub-code - Bypass Assignment Cannot Be Used (value: TBA2) to the downstream PLR to indicate that it cannot use the bypass tunnel assignment in the reverse direction. Upon receiving this error, the downstream PLR MAY remove the bypass tunnel assignment and select an alternate bypass tunnel if one available.

If multiple bypass tunnel assignments are present on the upstream PLR R6 at the time of a failure, any resulted asymmetry gets corrected using the re-coroute procedure after FRR as specified in Section 5.2.2 of this document.

5. Fast Reroute For Bidirectional GMPLS LSPs with In-band Signaling

When a bidirectional bypass tunnel is used, after a link failure, following procedure is followed when using the in-band signaling:

- o The downstream PLR reroutes protected LSP traffic and RSVP Path signaling over the bidirectional bypass tunnel using the procedures defined in [RFC4090]. The RSVP Path messages are modified as described in Section 6.4.3 of [RFC4090].
- o The upstream PLR reroutes protected LSP traffic upon detecting the link failure or upon receiving RSVP Path message over the bidirectional bypass tunnel.
- o The upstream PLR also reroutes protected LSP RSVP Resv signaling after receiving the modified RSVP Path message over the

bidirectional bypass tunnel. The upstream PLR uses the procedure defined in Section 7 of [RFC4090] to detect that RSVP Path messages have been rerouted over the bypass tunnel by the downstream PLR. The upstream PLR does not modify the RSVP Resv message before sending it over the bypass tunnel.

The above procedure allows both traffic and RSVP signaling to flow on symmetric paths in the forward and reverse directions of a protected bidirectional GMPLS LSP. The following sections describe the handling for link protection and node protection bypass tunnels.

5.1. Link Protection for Bidirectional GMPLS LSPs

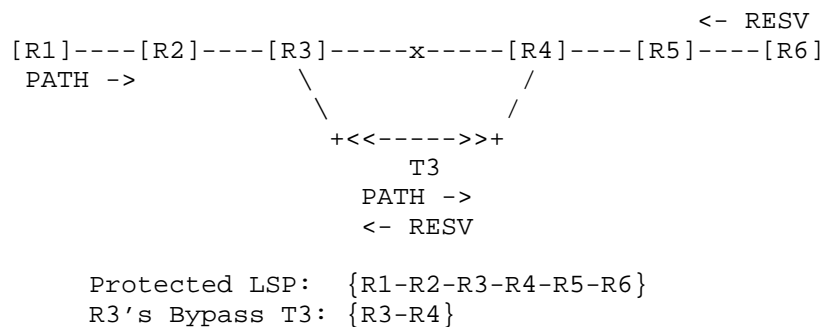


Figure 1: Flow of RSVP signaling after link failure and FRR

Consider the TE network shown in Figure 1. Assume every link in the network is protected with a link protection bypass tunnel (e.g., bypass tunnel T3). For the protected co-routed bidirectional LSP whose head-end is on node R1 and tail-end is on node R6, each traversed node (a potential PLR) assigns a link protection co-routed bidirectional bypass tunnel.

5.1.1. Behavior After Link Failure

Consider the link R3-R4 on the protected LSP path fails. The downstream PLR R3 and upstream PLR R4 independently trigger fast reroute to redirect traffic onto bypass tunnel T3 in the forward and reverse directions. The downstream PLR R3 also reroutes RSVP Path messages onto the bypass tunnel T3 using the procedures described in [RFC4090]. The upstream PLR R4 reroutes RSVP Resv messages onto the reverse bypass tunnel T3 upon receiving RSVP Path message over bypass tunnel T3.

5.1.2. Revertive Behavior After Fast Reroute

The revertive behavior defined in [RFC4090], Section 6.5.2, is

applicable to the link protection of bidirectional GMPLS LSPs. When using the local revertive mode, after the link R3-R4 (in Figure 1) is restored, following node behaviors apply:

- o The downstream PLR R3 starts sending the Path messages and traffic flow of the protected LSP over the restored link and stops sending them over the bypass tunnel.
- o The upstream PLR R4 starts sending the traffic flow of the protected LSP over the restored link and stops sending it over the bypass tunnel.
- o When upstream PLR R4 receives the protected LSP Path messages over the restored link, if not already done, it starts sending Resv messages and traffic flow of the protected LSP over the restored link and stops sending them over the bypass tunnel.

5.2. Node Protection for Bidirectional GMPLS LSPs

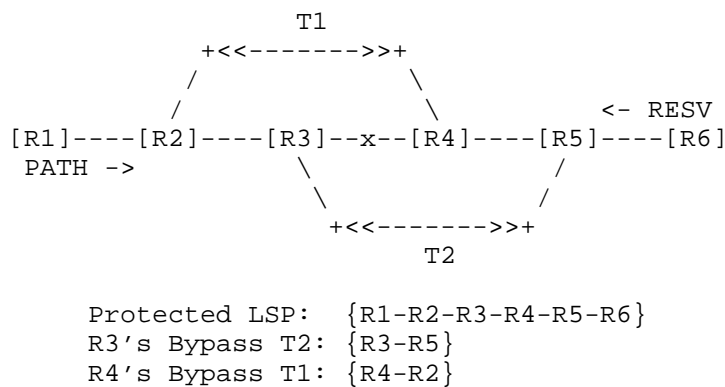


Figure 2: Flow of RSVP signaling after link failure and FRR

Consider the TE network shown in Figure 2. Assume every link in the network is protected with a node protection bypass tunnel. For the protected co-routed bidirectional LSP whose head-end is on node R1 and tail-end is on node R6, each traversed node (a potential PLR) assigns a node protection co-routed bidirectional bypass tunnel.

The solution introduces two phases to invoking FRR procedures by the PLR after the link failure. The first phase comprises of FRR procedures to fast reroute data traffic onto bypass tunnels in the forward and reverse directions. The second phase re-coroutes the data and signaling in the forward and reverse directions after the first phase.

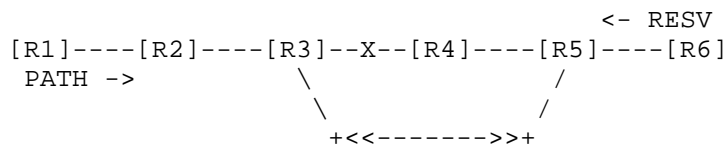
5.2.1. Behavior After Link Failure

Consider a link R3-R4 (in Figure 2) on the protected LSP path fails. The downstream PLR R3 and upstream PLR R4 independently trigger fast reroute procedures to redirect the protected LSP traffic onto respective bypass tunnels T2 and T1 in the forward and reverse directions. The downstream PLR R3 also reroutes RSVP Path messages over the bypass tunnel T2 using the procedures described in [RFC4090]. Note, at this point, node R4 stops receiving RSVP Path refreshes for the protected bidirectional LSP while protected traffic continues to flow over bypass tunnels. As node R4 does not receive Path messages over bypass tunnel T1, it does not reroute RSVP Resv messages over the reverse bypass tunnel T1.

5.2.2. Behavior After Link Failure To Re-coroute

The downstream MP R5 that receives rerouted protected LSP RSVP Path message through the bypass tunnel, in addition to the regular MP processing defined in [RFC4090], gets promoted to a Point of Remote Repair (PRR) role and performs the following actions to re-coroute signaling and data traffic over the same path in the reverse direction:

- o Finds the bypass tunnel in the reverse direction that terminates on the downstream PLR R3. Note: the downstream PLR R3's address can be extracted from the "IPV4 tunnel sender address" in the SENDER_TEMPLATE Object of the protected LSP (see [RFC4090], Section 6.1.1).
- o If reverse bypass tunnel is found and the protected LSP traffic is not already rerouted over the found bypass tunnel T2, the PRR R5 activates FRR reroute procedures to direct traffic over the found bypass tunnel T2 in the reverse direction. In addition, the PRR R5 also reroutes RSVP Resv over the bypass tunnel T2 in the reverse direction. This can happen when the downstream PLR has changed the bypass tunnel assignment but the upstream PLR has not yet processed the updated Path RRO and programmed the data-plane when link failure occurs.
- o If reverse bypass tunnel is not found, the PRR R5 immediately tears down the protected LSP.



Bypass Tunnel T2
traffic + signaling

Protected LSP: {R1-R2-R3-R4-R5-R6}
R3's Bypass T2: {R3-R5}

Figure 3: Flow of RSVP signaling after FRR and re-coroute

Figure 3 describes the path taken by the traffic and signaling after completing re-coroute of data and signaling in the forward and reverse paths described above. Node R4 will stop receiving the Path and Resv messages and it will timeout the RSVP soft-state, however, this will not cause the LSP to be torn down. RSVP signaling at node R2 is not affected by the FRR and re-corouting.

If downstream MP R5 receives multiple RSVP Path messages through multiple bypass tunnels (e.g., as a result of multiple failures), the PRR SHOULD identify a bypass tunnel that terminates on the farthest downstream PLR along the protected LSP path (closest to the protected bidirectional LSP head-end) and activate the reroute procedures mentioned above.

5.2.2.1. Re-coroute in Data-plane After Link Failure

The downstream MP (upstream PLR) MAY optionally support re-corouting in data-plane as follows. If the downstream MP has assigned a bidirectional bypass tunnel, as soon as the downstream MP receives the protected LSP packets on the bypass tunnel, it MAY switch the upstream traffic on to the bypass tunnel. In order to identify the protected LSP packets through the bypass tunnel, Penultimate Hop Popping (PHP) of the bypass tunnel MUST be disabled. The downstream MP checks whether the protected LSP signaling is rerouted over the found bypass tunnel, and if not, it performs the signaling procedure described in Section 5.2.2 of this document.

5.2.3. Revertive Behavior After Fast Reroute

The revertive behavior defined in [RFC4090], Section 6.5.2, is applicable to the node protection of bidirectional GMPLS LSPs. When using the local revertive mode, after the link R3-R4 (in Figures 2 and 3) is restored, following node behaviors apply:

- o The downstream PLR R3 starts sending the Path messages and traffic flow of the protected LSP over the restored link and stops sending them over the bypass tunnel.
- o The upstream PLR R4 (when the protected LSP is present) starts sending the traffic flow of the protected LSP over the restored

link towards downstream PLR R3 and forwarding the Path messages towards PRR R5 and stops sending the traffic over the bypass tunnel.

- o When upstream PLR R4 receives the protected LSP Path messages over the restored link, if not already done, the node R4 (when the protected LSP is present) starts sending Resv messages and traffic flow over the restored link towards downstream PLR R3 and forwarding the Path messages towards PRR R5 and stops sending them over the bypass tunnel.
- o When PRR R5 receives the protected LSP Path messages over the restored path, it starts sending Resv messages and traffic flow over the restored path and stops sending them over the bypass tunnel.

5.2.4. Behaviour After Node Failure

Consider the node R4 (in Figure 3) on the protected LSP path fails. The downstream PLR R3 and upstream PLR R5 independently trigger fast reroute procedures to redirect the protected LSP traffic onto bypass tunnel T2 in forward and reverse directions. The downstream PLR R3 also reroutes RSVP Path messages over the bypass tunnel T2 using the procedures described in [RFC4090]. The upstream PLR R5 reroutes RSVP Resv signaling after receiving the modified RSVP Path message over the bypass tunnel T2.

5.3. Unidirectional Link Failures

Unidirectional link failures can result in the traffic flowing on asymmetric paths in the forward and reverse directions. In addition, unidirectional link failures can cause RSVP soft-state timeout in the control-plane in some cases. As an example, if the unidirectional link failure is in the upstream direction (from R4 to R3 in Figures 1 and 2), the downstream PLR (node R3) can stop receiving the Resv messages of the protected LSP from the upstream PLR (node R4 in Figures 1 and 2) and this can cause RSVP soft-state timeout to occur on the downstream PLR (node R3).

A unidirectional link failure in the downstream direction (from R3 to R4 in Figures 1 and 2), does not cause RSVP soft-state timeout when using the FRR procedures defined in this document, since the upstream PLR (node R4 in Figure 1 and node R5 in Figure 2) triggers the re-coroute procedure (defined in Section 5.2.2 of this document) after receiving RSVP Path messages of the protected LSP over the bypass tunnel from the downstream PLR (node R3 in Figures 1 and 2).

6. Fast Reroute For Bidirectional GMPLS LSPs with Out-of-band Signaling

When using the GMPLS out-of-band signaling [RFC3473], after a link failure event, the RSVP messages are not rerouted over the bidirectional bypass tunnel by the downstream and upstream PLRs but instead rerouted over the control-channels to the downstream and upstream MPs, respectively.

The RSVP soft-state timeout after FRR as described in Section 5.2 of this document is equally applicable to the GMPLS out-of-band signaling as the RSVP signaling refreshes can stop reaching certain nodes along the protected LSP path after the downstream and upstream PLRs finish rerouting of the signaling messages. However, unlike with the in-band signaling, unidirectional link failures as described in Section 5.3 of this document do not result in soft-state timeout with GMPLS out-of-band signaling. Apart from this, the FRR procedure described in Section 5 of this document is equally applicable to the GMPLS out-of-band signaling.

7. Message and Object Definitions

7.1. BYPASS_ASSIGNMENT Subobject

The BYPASS_ASSIGNMENT subobject is used to inform the downstream MP of the bypass tunnel being assigned by the PLR. This can be used to coordinate the bypass tunnel assignment for the protected LSP by the downstream and upstream PLRs in the forward and reverse directions respectively prior or after the failure occurrence.

This subobject SHOULD be inserted into the Path RRO by the downstream PLR. It SHOULD NOT be inserted into an RRO by a node which is not a downstream PLR. It MUST NOT be changed by downstream LSRs and MUST NOT be added to a Resv RRO.

The BYPASS_ASSIGNMENT IPv4 subobject in RRO has the following format:

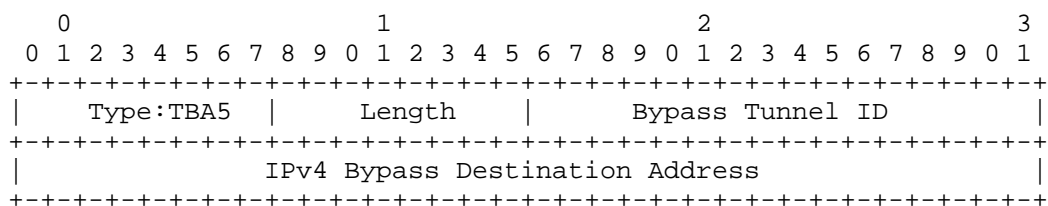


Figure 4: BYPASS ASSIGNMENT IPv4 RRO Subobject

Type

Downstream Bypass Assignment. Value is TBA5 by IANA.

Length

The Length contains the total length of the subobject in bytes, including the Type and Length fields. The length is 8 bytes.

Bypass Tunnel ID

The bypass tunnel identifier (16 bits).

Bypass Destination Address

The bypass tunnel IPv4 destination address.

The BYPASS_ASSIGNMENT IPv6 subobject in RRO has the following format:

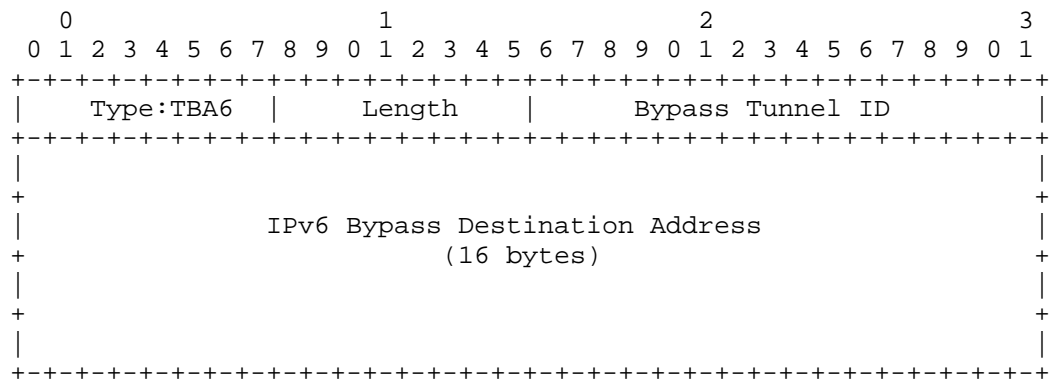


Figure 5: BYPASS_ASSIGNMENT IPv6 RRO Subobject

Type

Downstream Bypass Assignment. Value is TBA6 by IANA.

Length

The Length contains the total length of the subobject in bytes, including the Type and Length fields. The length is 20 bytes.

Bypass Tunnel ID

The bypass tunnel identifier (16 bits).

Bypass Destination Address

The bypass tunnel IPv6 destination address.

7.2. FRR Bypass Assignment Error Notify Message

New Error-code - FRR Bypass Assignment Error (value: TBA1) and its sub-codes are defined for the ERROR_SPEC Object (C-Type 6) [RFC2205] in this document, that is carried by the Notify message (Type 21) defined in [RFC3473] Section 4.3. This Error message is sent by the upstream PLR to the downstream PLR to notify a bypass assignment error. In the Notify message, the IP destination address is set to the node address of the downstream PLR that had initiated the bypass assignment. In the ERROR_SPEC Object, IP address is set to the node address of the upstream PLR that detected the bypass assignment error. This Error MUST NOT be sent in a Path Error message. This Error does not cause the protected LSP to be torn down.

8. Compatibility

New RSVP subobject BYPASS_ASSIGNMENT is defined for RECORD_ROUTE Object in this document that is carried in the RSVP Path message. Per [RFC3209], nodes not supporting this subobject will ignore the subobject but forward it without modification. As described in Section 7 of this document, this subobject is not carried in the RSVP Resv message and is ignored by sending the Notify message for FRR Bypass Assignment Error (with Subcode: Bypass Assignment Cannot Be Used) defined in this document. Nodes not supporting the Notify message defined in this document will ignore it but forward it without modification.

9. Security Considerations

This document introduces a new BYPASS_ASSIGNMENT subobject for the RECORD_ROUTE Object that is carried in an RSVP signaling message. Thus in the event of the interception of a signaling message, more information about LSP's fast reroute protection can be deduced than was previously the case. This is judged to be a very minor security risk as this information is already available by other means. If a MP does not find a matching bypass tunnel with given source and destination addresses locally, it ignores the BYPASS_ASSIGNMENT subobject. Due to this, security risk introduced by inserting a random address in this subobject is minimal. The Notify message for FRR Bypass Assignment Error defined in this document does not result in tear-down of the protected LSP and is not service affecting.

Security considerations for RSVP-TE and GMPLS signaling extensions

are covered in [RFC3209] and [RFC3473]. Further, general considerations for securing RSVP-TE in MPLS-TE and GMPLS networks can be found in [RFC5920]. This document updates the mechanisms defined in [RFC4090], which also discusses related security measures and are also applicable to this document. As specified in [RFC4090], a PLR and its selected merge point trust RSVP messages received from each other. The security considerations pertaining to the original RSVP protocol [RFC2205] also remain relevant to the updates in this document.

10. IANA Considerations

10.1. BYPASS_ASSIGNMENT Subobject

IANA manages the "RSVP PARAMETERS" registry located at <<http://www.iana.org/assignments/rsvp-parameters>>. IANA is requested to assign a value for the new BYPASS_ASSIGNMENT subobject in the "Class Type 21 ROUTE_RECORD - Type 1 Route Record" registry.

This document introduces a new subobject for RECORD_ROUTE Object:

Type	Description	Carried in Path	Carried in Resv	Reference
TBA5 By IANA	BYPASS_ASSIGNMENT IPv4 subobject	Yes	No	This document
TBA6 By IANA	BYPASS_ASSIGNMENT IPv6 subobject	Yes	No	This document

10.2. FRR Bypass Assignment Error Notify Message

IANA maintains the "Resource Reservation Protocol (RSVP) Parameters" registry (see <<http://www.iana.org/assignments/rsvp-parameters>>). The "Error Codes and Globally-Defined Error Value Sub-Codes" subregistry is included in this registry.

This registry has been extended for the new Error-code and Sub-codes defined in this document as follows:

- o Error-code TBA1: FRR Bypass Assignment Error
- o Sub-code TBA2: Bypass Assignment Cannot Be Used

- o Sub-code TBA3: Bypass Tunnel Not Found
- o Sub-code TBA4: One-to-one Bypass Already In-use

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.
- [RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.
- [RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, May 2005.
- [RFC4561] Vasseur, J.P., Ed., Ali, Z., and S. Sivabalan, "Definition of a Record Route Object (RRO) Node-Id Sub-Object", RFC 4561, June 2006.

11.2. Informative References

- [RFC3471] Berger, L., Editor, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, January 2003.
- [RFC4990] Shiimoto, K., Papneja, R., and R. Rabbat, "Use of Addresses in Generalized Multiprotocol Label Switching (GMPLS) Networks", RFC 4990, September 2007.
- [RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.
- [RFC6378] Weingarten, Y., Bryant, S., Osborne, E., Sprecher, N., and A. Fulignoli, "MPLS Transport Profile (MPLS-TP) Linear Protection", RFC 6378, October 2011.
- [RFC7551] Zhang, F., Ed., Jing, R., and Gandhi, R., Ed., "RSVP-TE Extensions for Associated Bidirectional LSPs", RFC 7551, May 2015.

Acknowledgements

Authors would like to thank George Swallow for many useful comments and suggestions. Authors would like to thank Lou Berger for the guidance on this work and for providing review comments. Authors would also like to thank Nobo Akiya, Loa Andersson, Matt Hartley, Himanshu Shah, Gregory Mirsky, Mach Chen, Vishnu Pavan Beeram and Alia Atlas for reviewing this document and providing valuable comments. A special thanks to Adrian Farrel for his thorough review of this document.

Contributors

Frederic Jounay
Orange
CH

EMail: frederic.jounay@salt.ch

Lizhong Jin
Shanghai
CN

EMail: lizho.jin@gmail.com

Authors' Addresses

Mike Taillon
Cisco Systems, Inc.

EMail: mtaillon@cisco.com

Tarek Saad (editor)
Cisco Systems, Inc.

EMail: tsaad@cisco.com

Rakesh Gandhi (editor)
Cisco Systems, Inc.

EMail: rgandhi@cisco.com

Zafar Ali
Cisco Systems, Inc.

EMail: zali@cisco.com

Manav Bhatia
Nokia
Bangalore, India

EMail: manav.bhatia@nokia.com

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

H. Chen
Huawei Technologies
A. Liu
Ciena
T. Saad
Cisco Systems
F. Xu
Verizon
L. Huang
China Mobile
March 18, 2018

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

Abstract

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

Status of this Memo

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

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

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

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

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. Egress Local Protection	3
2. Conventions Used in This Document	4
3. Terminologies	4
4. Protocol Extensions	5
4.1. Extensions to SERO	5
4.1.1. Primary Egress Subobject	7
4.1.2. P2P LSP ID Subobject	8
5. Egress Protection Behaviors	9
5.1. Ingress Behavior	9
5.2. Primary Egress Behavior	10
5.3. Backup Egress Behavior	10
5.4. Transit Node and PLR Behavior	11
5.4.1. Signaling for One-to-One Protection	12
5.4.2. Signaling for Facility Protection	12
5.4.3. Signaling for S2L Sub LSP Protection	13
5.4.4. PLR Procedures during Local Repair	13
6. Application Traffic Considerations	14
6.1. A Typical Application	14
6.2. PLR Procedure for Applications	16
6.3. Egress Procedures for Applications	17
7. Security Considerations	17
8. IANA Considerations	17
9. Co-authors and Contributors	18
10. Acknowledgement	19
11. References	19
11.1. Normative References	19
11.2. Informative References	20
Authors' Addresses	20

Figure 1: Backup LSP for Locally Protecting Egress

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

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

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

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

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

2. Conventions Used in This Document

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

3. Terminologies

The following terminologies are used in this document.

LSP: Label Switched Path

TE: Traffic Engineering

P2MP: Point-to-MultiPoint
P2P: Point-to-Point
LSR: Label Switching Router
RSVP: Resource ReSerVation Protocol
S2L: Source-to-Leaf
SERO: Secondary Explicit Route Object
RRO: Record Route Object
BFD: Bidirectional Forwarding Detection
VPN: Virtual Private Network
L3VPN: Layer 3 VPN
VRF: Virtual Routing and Forwarding
LFIB: Label Forwarding Information Base
UA: Upstream Assigned
PLR: Point of Local Repair
BGP: Border Gateway Protocol
CE: Customer Edge
PE: Provider Edge

4. Protocol Extensions

4.1. Extensions to SERO

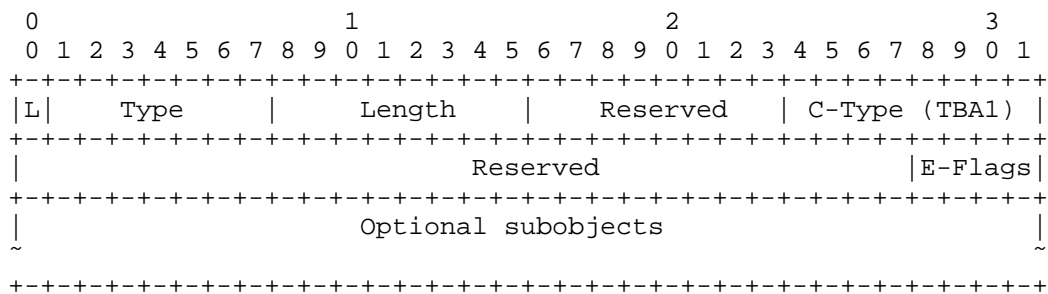
The Secondary Explicit Route object (SERO) is defined in RFC 4873. The format of the SERO is re-used.

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

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

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

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



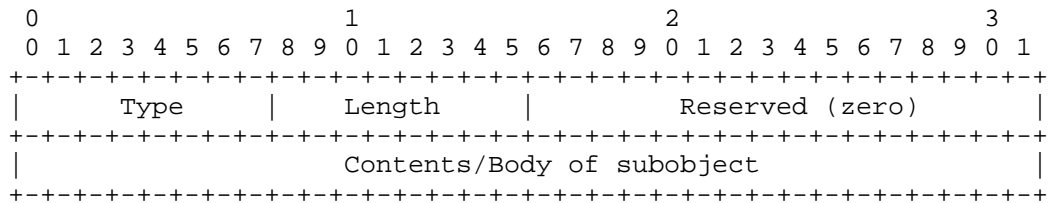
E-Flags are defined for egress local protection.

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

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

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

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



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

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

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

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

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

4.1.1. Primary Egress Subobject

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

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

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               IPv4 address (4 bytes)                       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

- o IPv4 address: IPv4 address of the primary egress node

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

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               IPv6 address (16 bytes)                       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

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

4.1.2. P2P LSP ID Subobject

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

4.1.2.1. IPv4 P2P LSP ID Subobject

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

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               P2P LSP Tunnel Egress IPv4 Address             |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Reserved (MUST be zero)         |   Tunnel ID                           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Extended Tunnel ID                           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

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

4.1.2.2. IPv6 P2P LSP ID Subobject

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

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
~                P2P LSP Tunnel Egress IPv6 Address (16 bytes)                ~
+-----+-----+-----+-----+-----+-----+-----+-----+
|   Reserved (MUST be zero)   |           Tunnel ID           |
+-----+-----+-----+-----+-----+-----+-----+-----+
~                Extended Tunnel ID (16 bytes)                ~
+-----+-----+-----+-----+-----+-----+-----+-----+

```

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

5. Egress Protection Behaviors

5.1. Ingress Behavior

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

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

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

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

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

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

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

5.2. Primary Egress Behavior

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

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

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

5.3. Backup Egress Behavior

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

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

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

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

5.4. Transit Node and PLR Behavior

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

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

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

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

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

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

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

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

5.4.1. Signaling for One-to-One Protection

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

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

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

5.4.2. Signaling for Facility Protection

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

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

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

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

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

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

5.4.3. Signaling for S2L Sub LSP Protection

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

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

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

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

5.4.4. PLR Procedures during Local Repair

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

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

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

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

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

6. Application Traffic Considerations

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

6.1. A Typical Application

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

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

to Figure 2). This solution entailed:

1. A multi-hop BFD session between ingress R1 and egress node L1 of primary LSP. The BFD session is represented by dots (....).
2. A backup LSP from ingress R1 to backup egress node La, which is indicated by ands (&&&&).
3. La sends R1 a VPN backup label and related information via BGP.
4. R1 has a VRF with two sets of routes for CE2: one set uses the primary LSP and L1 as next hop; the other uses the backup LSP and La as next hop.

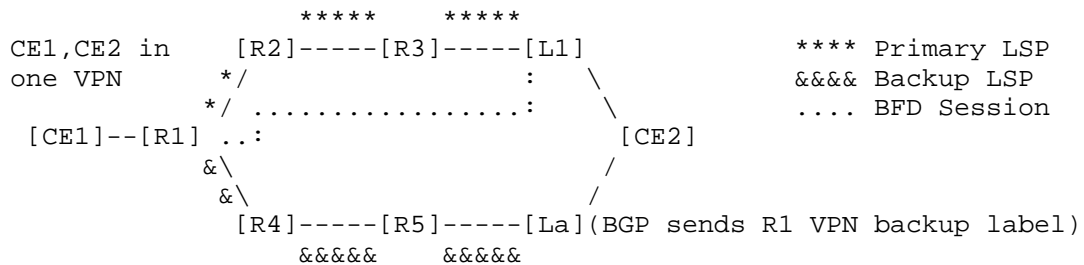


Figure 2: Protect Egress for L3VPN Traffic

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

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

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

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

2. A backup LSP from R3 to backup egress node La. This is different from the backup LSP in Figure 2, which is an end to end LSP from ingress R1 to backup egress node La.
3. Primary egress node L1 sends backup egress node La the VPN label as UA label and related information. The backup egress node La uses the backup LSP label as a context label and creates a forwarding entry using the VPN label in a LFIB for the primary egress node L1.
4. L1 and La is virtualized as one node (or address). R1 has a VRF with one set of routes for CE2, using the primary LSP from R1 to L1 and virtualized node as next hop. This can be achieved by configuring a same local address on L1 and La, using the address as a destination of the LSP and BGP next hop for the VPN traffic. The cost to L1 is configured to be less than the cost to La.

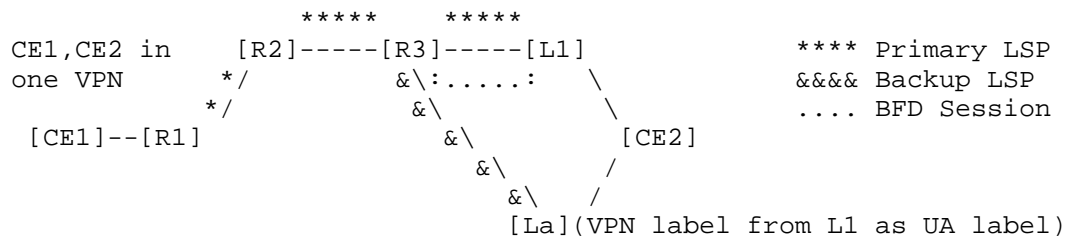


Figure 3: Locally Protect Egress for L3VPN Traffic

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

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

6.2. PLR Procedure for Applications

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

6.3. Egress Procedures for Applications

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

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

7. Security Considerations

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

8. IANA Considerations

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

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

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

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

9. Co-authors and Contributors

1. Co-authors

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

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

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

Zhenbin Li
Huawei Technologies
Email: lizhenbin@huawei.com

2. Contributors

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

Nan Meng
Huawei Technologies
Email: mengnan@huawei.com

Prejeeth Kaladharan
Huawei Technologies
Email: prejeeth@gmail.com

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

10. Acknowledgement

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

11. References

11.1. Normative References

- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, DOI 10.17487/RFC4090, May 2005, <<https://www.rfc-editor.org/info/rfc4090>>.
- [RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <<https://www.rfc-editor.org/info/rfc4875>>.
- [RFC4873] Berger, L., Bryskin, I., Papadimitriou, D., and A. Farrel, "GMPLS Segment Recovery", RFC 4873, DOI 10.17487/RFC4873, May 2007, <<https://www.rfc-editor.org/info/rfc4873>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

11.2. Informative References

- [RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <<https://www.rfc-editor.org/info/rfc2205>>.
- [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <<https://www.rfc-editor.org/info/rfc5331>>.
- [RFC4872] Lang, J., Ed., Rekhter, Y., Ed., and D. Papadimitriou, Ed., "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery", RFC 4872, DOI 10.17487/RFC4872, May 2007, <<https://www.rfc-editor.org/info/rfc4872>>.
- [RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <<https://www.rfc-editor.org/info/rfc3473>>.
- [FRAMEWK] Shen, Y., Jeyanthan, M., Decraene, B., and H. Gredler, "MPLS Egress Protection Framework", draft-shen-mpls-egress-protection-framework , October 2016.

Authors' Addresses

Huaimo Chen
Huawei Technologies
Boston, MA
USA

Email: huaimo.chen@huawei.com

Autumn Liu
Ciena
USA

Email: hliu@ciena.com

Tarek Saad
Cisco Systems
Email: tsaad@cisco.com

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

Lu Huang
China Mobile
No.32 Xuanwumen West Street, Xicheng District
Beijing, 100053
China
Email: huanglu@chinamobile.com

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

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

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

Abstract

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

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 11 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
2. Requirements Language	3
2.1. Prefixes in Data Node Names	4
2.2. Model Tree Diagrams	4
3. Design Considerations	5
3.1. State Data Organization	5
4. Model Overview	6
4.1. Module Relationship	6
5. TE YANG Model	7
5.1. Module Structure	7
5.1.1. TE Globals	9
5.1.2. TE Tunnels	12
5.1.3. TE LSPs	19
5.2. Tree Diagram	19
5.3. YANG Module	60
6. TE Device YANG Model	98
6.1. Module Structure	99
6.1.1. TE Interfaces	99
6.2. Tree Diagram	100
6.3. YANG Module	102
7. Notifications	116
8. TE Generic and Helper YANG Modules	117
9. IANA Considerations	117
10. Security Considerations	117
11. Acknowledgement	119
12. Contributors	119
13. Appendix A: Data Tree Examples	119
13.1. Basic Tunnel Setup	120
13.2. Global Named Path Constraints	121
13.3. Tunnel with Global Path Constraint	121
13.4. Tunnel with Per-tunnel Path Constraint	122
13.5. Tunnel State	123

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

1. Introduction

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

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

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

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

2. Requirements Language

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

The following terms are defined in [RFC6241] and are used in this specification:

- * client
- * configuration data

- * state data

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

- * augment

- * data model

- * data node

2.1. Prefixes in Data Node Names

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

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

Table 1: Prefixes and corresponding YANG modules

2.2. Model Tree Diagrams

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

3. Design Considerations

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

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

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

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

- * The generic TE YANG data model 'ietf-te' contains device independent data and can be used to model data off a device (e.g. on a TE controller). The device-specific TE data is defined in module 'ietf-te-device' as shown in Figure 1,
- * In general, minimal elements in the model are designated as "mandatory" to allow freedom to vendors to adapt the data model to their specific product implementation.
- * Suitable defaults are specified for all configurable elements.
- * The model declares a number of TE functions as features that can be optionally supported.

3.1. State Data Organization

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

4. Model Overview

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

4.1. Module Relationship

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

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

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

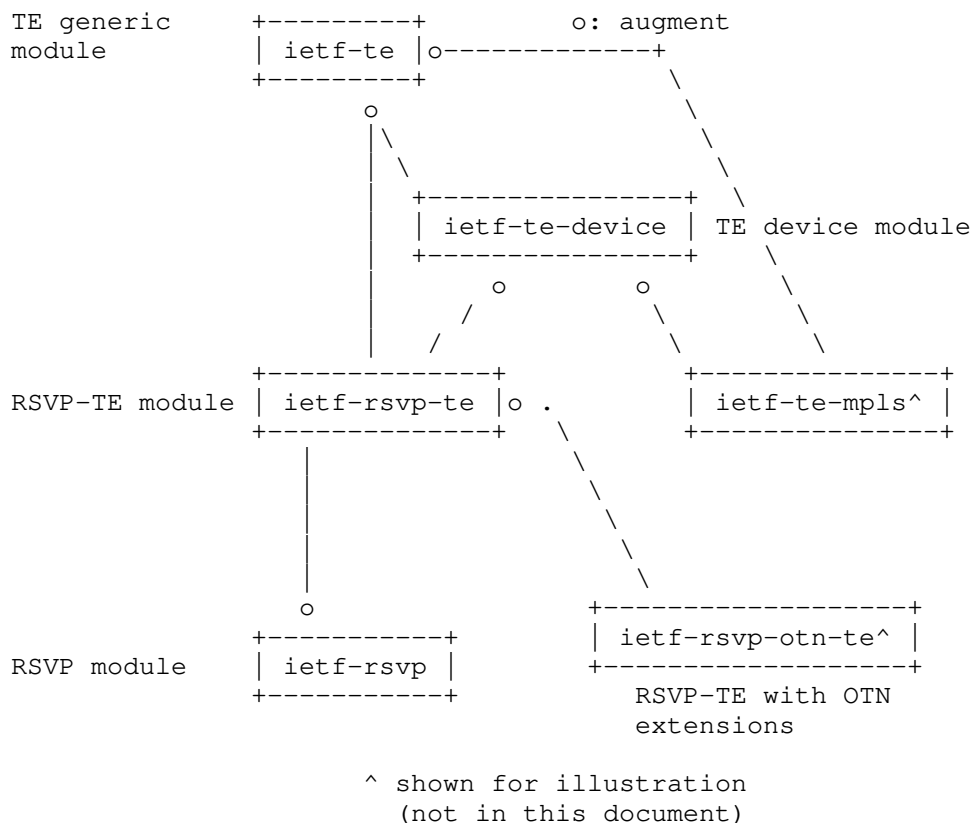


Figure 1: Relationship of TE module(s) with signaling protocol modules

5. TE YANG Model

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

The detailed tree structure is provided in Figure 2.

5.1. Module Structure

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

globals:

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

tunnels:

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

lsps:

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

tunnels-path-compute:

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

tunnels-action:

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

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

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

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

5.1.1. TE Globals

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

```

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

```

Figure 3: TE globals YANG subtree high-level structure

named-admin-groups:

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

named-srlgs:

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

named-path-constraints:

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

```

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

```

Figure 4: Named path constraints YANG subtree

- o te-bandwidth: A YANG container that holds the technology agnostic TE bandwidth constraint.
- o link-protection: A YANG leaf that holds the link protection type constraint required for the links to be included in the computed path.
- o setup/hold priority: A YANG leaf that holds the LSP setup and hold admission priority as defined in [RFC3209].
- o signaling-type: A YANG leaf that holds the LSP setup type, such as RSVP-TE or SR.
- o path-metric-bounds: A YANG container that holds the set of metric bounds applicable on the computed TE tunnel path.

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

The `'route-object-include-exclude'` is used to configure constraints on which route objects (e.g., nodes, links) are included or excluded in the path computation.

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

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

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

5.1.2. TE Tunnels

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

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

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

TE Tunnel:

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

TE Tunnel Segment:

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

```

+--rw tunnels
|   +--rw tunnel* [name]
|   |   +--rw name                               string
|   |   +--rw alias?                             string
|   |   +--rw identifier?                         uint32
|   |   +--rw color?                             uint32
|   |   +--rw description?                       string
|   |   +--ro operational-state?                 identityref
|   |   +--rw encoding?                         identityref
|   |   +--rw switching-type?                   identityref
|   |   +--rw admin-state?                     identityref
|   |   +--rw reoptimize-timer?                 uint16
|   |   +--rw source?                           te-types:te-node-id
|   |   +--rw destination?                     te-types:te-node-id
|   |   +--rw src-tunnel-tp-id?                 binary
|   |   +--rw dst-tunnel-tp-id?                 binary
|   |   +--rw controller
|   |   |   +--rw protocol-origin?               identityref
|   |   |   +--rw controller-entity-id?         string
|   |   +--rw bidirectional?                   boolean
|   |   +--rw association-objects
|   |   |   +--rw association-object* [association-key]
|   |
|   |   // ..
|   |   |
|   |   +--rw protection
|   |
|   |   // ..
|   |   |
|   |   +--rw restoration
|   |
|   |   // ..
|   |   |
|   |   +--rw te-topology-identifier
|   |
|   |   // ..
|   |   |
|   |   +--rw hierarchy
|   |
|   |   // ..

```

Figure 5: TE Tunnel list YANG subtree structure

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

operational-state:

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

name:

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

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

alias:

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

identifier:

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

color:

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

encoding/switching:

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

reoptimize-timer:

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

source/destination:

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

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

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

controller:

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

bidirectional:

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

association-objects:

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

protection:

A YANG container that holds the TE Tunnel protection properties.

restoration:

A YANG container that holds the TE Tunnel restoration properties.

te-topology-identifier:

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

```

+--rw hierarchy
|   +--rw dependency-tunnels
|   |   +--rw dependency-tunnel* [name]
|   |   |   +--rw name
|   |   |   |   -> ../../../../tunnels/tunnel/name
|   |   |   +--rw encoding?          identityref
|   |   |   +--rw switching-type?    identityref
|   |   +--rw hierarchical-link
|   |   |   +--rw local-te-node-id?    te-types:te-node-id
|   |   |   +--rw local-te-link-tp-id? te-types:te-tp-id
|   |   |   +--rw remote-te-node-id?   te-types:te-node-id
|   |   +--rw te-topology-identifier
|   |   |   +--rw provider-id?    te-global-id
|   |   |   +--rw client-id?     te-global-id
|   |   |   +--rw topology-id?   te-topology-id

```

Figure 6: TE Tunnel hierarchy YANG subtree

hierarchy:

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

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

5.1.2.1. TE Tunnel Paths

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

```

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

```

Figure 7: TE Tunnel paths YANG tree structure

primary-paths:

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

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

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

secondary-paths:

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

secondary-reverse-paths:

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

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

compute-only:

A path of TE Tunnel is, by default, provisioned so that it can be instantiated in forwarding to carry traffic as soon as a valid path is computed. In some cases, a TE path may be provisioned for the only purpose of computing a path and reporting it without the need to instantiate the LSP or commit any resources. In such a case, the path is configured in 'compute-only' mode to distinguish it from the default behavior. A 'compute-only' path is configured as usual with the associated per path constraint(s) and properties on a device or TE controller. The device or TE controller computes the feasible path(s) subject to configured constraints. A client may query the 'compute-only' computed path properties 'on-demand', or alternatively, can subscribe to be notified of computed path(s) and whenever the path properties change.

use-path-computation:

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

lockdown:

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

te-topology-identifier:

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

optimizations:

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

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

computed-path-error-infos:

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

lsps:

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

5.1.3. TE LSPs

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

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

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

5.2. Tree Diagram

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

```

module: ietf-te
+--rw te!
  +--rw globals
    +--rw named-admin-groups
      +--rw named-admin-group* [name]
        {te-types:extended-admin-groups,te-types:named-extend
ed-admin-groups}?
        +--rw name string
        +--rw bit-position? uint32
    +--rw named-srlgs
      +--rw named-srlg* [name] {te-types:named-srlg-groups}?
        +--rw name string
        +--rw value? te-types:srlg
        +--rw cost? uint32
    +--rw named-path-constraints
      +--rw named-path-constraint* [name]
        {te-types:named-path-constraints}?
        +--rw name string
        +--rw te-bandwidth
          +--rw (technology)?
            +--:(generic)
              +--rw generic? te-bandwidth
        +--rw link-protection? identityref
        +--rw setup-priority? uint8
        +--rw hold-priority? uint8
        +--rw signaling-type? identityref
        +--rw path-metric-bounds
          +--rw path-metric-bound* [metric-type]
            +--rw metric-type identityref
            +--rw upper-bound? uint64
        +--rw path-affinities-values
          +--rw path-affinities-value* [usage]
            +--rw usage identityref
            +--rw value? admin-groups
        +--rw path-affinity-names
          +--rw path-affinity-name* [usage]
            +--rw usage identityref
            +--rw affinity-name* [name]
              +--rw name string
        +--rw path-srlgs-lists
          +--rw path-srlgs-list* [usage]
            +--rw usage identityref
            +--rw values* srlg
        +--rw path-srlgs-names
          +--rw path-srlgs-name* [usage]
            +--rw usage identityref
            +--rw names* string
        +--rw disjointness?

```

```

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

```

```

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

```



```

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

```

```

+--rw id?                               uint16
+--rw source
  +--rw id?      te-gen-node-id
  +--rw type?    enumeration
+--rw association-object-extended* [association-key]
  +--rw association-key    string
  +--rw type?              identityref
  +--rw id?                uint16
  +--rw source
    +--rw id?      te-gen-node-id
    +--rw type?    enumeration
  +--rw global-source?    uint32
  +--rw extended-id?      yang:hex-string
+--rw protection
  +--rw enable?                                boolean
  +--rw protection-type?                       identityref
  +--rw protection-reversion-disable?          boolean
  +--rw hold-off-time?                         uint32
  +--rw wait-to-revert?                       uint16
  +--rw aps-signal-id?                        uint8
+--rw restoration
  +--rw enable?                                boolean
  +--rw restoration-type?                     identityref
  +--rw restoration-scheme?                   identityref
  +--rw restoration-reversion-disable?          boolean
  +--rw hold-off-time?                       uint32
  +--rw wait-to-restore?                     uint16
  +--rw wait-to-revert?                     uint16
+--rw te-topology-identifier
  +--rw provider-id?    te-global-id
  +--rw client-id?      te-global-id
  +--rw topology-id?    te-topology-id
+--rw te-bandwidth
  +--rw (technology)?
    +--:(generic)
      +--rw generic?    te-bandwidth
+--rw link-protection?                identityref
+--rw setup-priority?                  uint8
+--rw hold-priority?                   uint8
+--rw signaling-type?                  identityref
+--rw hierarchy
  +--rw dependency-tunnels
    +--rw dependency-tunnel* [name]
      +--rw name
        -> /te/tunnels/tunnel/name
      +--rw encoding?            identityref
      +--rw switching-type?      identityref
+--rw hierarchical-link

```

```

+--rw local-te-node-id?          te-types:te-node-id
+--rw local-te-link-tp-id?       te-types:te-tp-id
+--rw remote-te-node-id?         te-types:te-node-id
+--rw te-topology-identifier
  +--rw provider-id?             te-global-id
  +--rw client-id?               te-global-id
  +--rw topology-id?             te-topology-id
+--rw primary-paths
  +--rw primary-path* [name]
    +--rw name                    string
    +--rw path-computation-method? identityref
    +--rw path-computation-server
      +--rw id?                   te-gen-node-id
      +--rw type?                 enumeration
    +--rw compute-only?           empty
    +--rw use-path-computation?   boolean
    +--rw lockdown?               empty
    +--rw path-scope?             identityref
    +--rw preference?             uint8
    +--rw k-requested-paths?      uint8
    +--rw association-objects
      +--rw association-object* [association-key]
        +--rw association-key     string
        +--rw type?               identityref
        +--rw id?                 uint16
        +--rw source
          +--rw id?               te-gen-node-id
          +--rw type?             enumeration
      +--rw association-object-extended*
        [association-key]
        +--rw association-key     string
        +--rw type?               identityref
        +--rw id?                 uint16
        +--rw source
          +--rw id?               te-gen-node-id
          +--rw type?             enumeration
        +--rw global-source?      uint32
        +--rw extended-id?        yang:hex-string
+--rw optimizations
  +--rw (algorithm)?
    +--:(metric) {path-optimization-metric}?
      +--rw optimization-metric* [metric-type]
        +--rw metric-type
          identityref
        +--rw weight?
          uint8
        +--rw explicit-route-exclude-objects
        +--rw route-object-exclude-object*

```

```

[index]
+---rw index
|      uint32
+---rw (type)?
+---:(numbered-node-hop)
|   +---rw numbered-node-hop
|   |   +---rw node-id
|   |   |   te-node-id
|   |   +---rw hop-type?
|   |       te-hop-type
+---:(numbered-link-hop)
|   +---rw numbered-link-hop
|   |   +---rw link-tp-id
|   |   |   te-tp-id
|   |   +---rw hop-type?
|   |   |   te-hop-type
|   |   +---rw direction?
|   |       te-link-direction
+---:(unnumbered-link-hop)
|   +---rw unnumbered-link-hop
|   |   +---rw link-tp-id
|   |   |   te-tp-id
|   |   +---rw node-id
|   |   |   te-node-id
|   |   +---rw hop-type?
|   |   |   te-hop-type
|   |   +---rw direction?
|   |       te-link-direction
+---:(as-number)
|   +---rw as-number-hop
|   |   +---rw as-number
|   |   |   inet:as-number
|   |   +---rw hop-type?
|   |       te-hop-type
+---:(label)
|   +---rw label-hop
|   |   +---rw te-label
|   |   |   +---rw (technology)?
|   |   |   |   +---:(generic)
|   |   |   |   |   +---rw generic?
|   |   |   |       rt-types:ge
|   |   |   |       +---rw direction?
|   |   |   |       |   te-label-directio
+---:(srlg)
|   +---rw srlg
|   |   +---rw srlg?   uint32

```

```

+---rw explicit-route-include-objects
+---rw route-object-include-object*
    [index]
+---rw index
    |   uint32
+---rw (type)?
+---:(numbered-node-hop)
    +---rw numbered-node-hop
        +---rw node-id
            |   te-node-id
        +---rw hop-type?
            |   te-hop-type
+---:(numbered-link-hop)
    +---rw numbered-link-hop
        +---rw link-tp-id
            |   te-tp-id
        +---rw hop-type?
            |   te-hop-type
        +---rw direction?
            |   te-link-direction
+---:(unnumbered-link-hop)
    +---rw unnumbered-link-hop
        +---rw link-tp-id
            |   te-tp-id
        +---rw node-id
            |   te-node-id
        +---rw hop-type?
            |   te-hop-type
        +---rw direction?
            |   te-link-direction
+---:(as-number)
    +---rw as-number-hop
        +---rw as-number
            |   inet:as-number
        +---rw hop-type?
            |   te-hop-type
+---:(label)
    +---rw label-hop
        +---rw te-label
            +---rw (technology)?
                +---:(generic)
                    +---rw generic?
                        |   rt-types:ge
neralized-label
n
+---rw tiebreakers

```

```

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

```

bel

```

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

```

bel

```

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

```



```

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

```

```

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

```

```

    +--ro error-description?  string
    +--ro error-timestamp?    yang:date-and-time
    +--ro error-node-id?      te-types:te-node-id
    +--ro error-link-id?      te-types:te-tp-id
    +--ro lsp-id?             uint16
+--ro lsps
  +--ro lsp* [node lsp-id]
    +--ro tunnel-name?
      |      -> /te/lsps/lsp/tunnel-name
    +--ro node          -> /te/lsps/lsp/node
    +--ro lsp-id        -> /te/lsps/lsp/lsp-id
+--rw primary-reverse-path
  +--rw name?                                string
  +--rw path-computation-method?
    |      identityref
  +--rw path-computation-server
    |      +--rw id?      te-gen-node-id
    |      +--rw type?    enumeration
  +--rw compute-only?                        empty
  +--rw use-path-computation?
    |      boolean
  +--rw lockdown?                            empty
  +--ro path-scope?
    |      identityref
  +--rw association-objects
    +--rw association-object* [association-key]
      |      +--rw association-key  string
      |      +--rw type?            identityref
      |      +--rw id?              uint16
      |      +--rw source
      |        |      +--rw id?      te-gen-node-id
      |        |      +--rw type?    enumeration
    +--rw association-object-extended*
      |      [association-key]
      |      +--rw association-key  string
      |      +--rw type?            identityref
      |      +--rw id?              uint16
      |      +--rw source
      |        |      +--rw id?      te-gen-node-id
      |        |      +--rw type?    enumeration
      |      +--rw global-source?   uint32
      |      +--rw extended-id?     yang:hex-string
  +--rw optimizations
    +--rw (algorithm)?
      +--:(metric) {path-optimization-metric}?
        |      +--rw optimization-metric* [metric-type]
        |        |      +--rw metric-type
        |        |      |      identityref

```

```

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

```

tion

```

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

```

```

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

```

```

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

```

					<pre> +---rw direction? te-link-direction +---:(unnumbered-link-hop) +---rw unnumbered-link-hop +---rw link-tp-id te-tp-id +---rw node-id te-node-id +---rw hop-type? te-hop-type +---rw direction? te-link-direction +---:(as-number) +---rw as-number-hop +---rw as-number inet:as-number +---rw hop-type? te-hop-type +---:(label) +---rw label-hop +---rw te-label +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized </pre>
-label					
					<pre> +---rw direction? te-label-direction +---:(srlg) +---rw srlg +---rw srlg? uint32 +---rw path-in-segment! +---rw label-restrictions +---rw label-restriction* [index] +---rw restriction? enumeration +---rw index uint32 +---rw label-start +---rw te-label +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized-la </pre>
bel					
					<pre> +---rw direction? te-label-direction +---rw label-end +---rw te-label +---rw (technology)? +---:(generic) +---rw generic? rt-types:generalized-la </pre>
bel					
					<pre> +---rw direction? </pre>

					<pre> te-label-direction +--rw label-step +--rw (technology)? +--:(generic) +--rw generic? int32 +--rw range-bitmap? yang:hex-string +--rw path-out-segment! +--rw label-restrictions +--rw label-restriction* [index] +--rw restriction? enumeration +--rw index uint32 +--rw label-start +--rw te-label +--rw (technology)? +--:(generic) +--rw generic? rt-types:generalized-la </pre>
bel					
					<pre> +--rw direction? te-label-direction +--rw label-end +--rw te-label +--rw (technology)? +--:(generic) +--rw generic? rt-types:generalized-la </pre>
bel					
					<pre> +--rw direction? te-label-direction +--rw label-step +--rw (technology)? +--:(generic) +--rw generic? int32 +--rw range-bitmap? yang:hex-string +--ro computed-paths-properties +--ro computed-path-properties* [k-index] +--ro k-index uint8 +--ro path-properties +--ro path-metric* [metric-type] +--ro metric-type identityref +--ro accumulative-value? uint64 +--ro path-affinities-values +--ro path-affinities-value* [usage] +--ro usage identityref +--ro value? admin-groups +--ro path-affinity-names +--ro path-affinity-name* [usage] </pre>

```

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

```

```

+--ro te-label
+--ro (technology)?
+--:(generic)
+--ro generic?
rt-types:ge

neralized-label

+--ro direction?
te-label-directio

n

+--ro te-bandwidth
+--ro (technology)?
+--:(generic)
+--ro generic? te-bandwidth
+--ro disjointness-type?
te-types:te-path-disjointness
+--ro computed-path-error-infos
+--ro computed-path-error-info* []
+--ro error-description? string
+--ro error-timestamp?
| yang:date-and-time
+--ro error-reason? identityref
+--ro lsp-provisioning-error-infos
+--ro lsp-provisioning-error-info* []
+--ro error-description? string
+--ro error-timestamp?
| yang:date-and-time
+--ro error-node-id?
| te-types:te-node-id
+--ro error-link-id?
| te-types:te-tp-id
+--ro lsp-id? uint16
+--ro lsps
+--ro lsp* [node lsp-id]
+--ro tunnel-name?
| -> /te/lsps/lsp/tunnel-name
+--ro node -> /te/lsps/lsp/node
+--ro lsp-id -> /te/lsps/lsp/lsp-id
+--rw candidate-secondary-reverse-paths
+--rw candidate-secondary-reverse-path*
| [secondary-path]
+--rw secondary-path leafref
+--rw candidate-secondary-paths
+--rw candidate-secondary-path* [secondary-path]
+--rw secondary-path leafref
+--ro active? boolean
+--rw secondary-paths
+--rw secondary-path* [name]
+--rw name string

```

```

+--rw path-computation-method?          identityref
+--rw path-computation-server
|   +--rw id?        te-gen-node-id
|   +--rw type?      enumeration
+--rw compute-only?                      empty
+--rw use-path-computation?              boolean
+--rw lockdown?                          empty
+--ro path-scope?                        identityref
+--rw preference?                        uint8
+--rw association-objects
|   +--rw association-object* [association-key]
|   |   +--rw association-key      string
|   |   +--rw type?                identityref
|   |   +--rw id?                  uint16
|   |   +--rw source
|   |   |   +--rw id?        te-gen-node-id
|   |   |   +--rw type?      enumeration
|   +--rw association-object-extended*
|   |   [association-key]
|   |   +--rw association-key      string
|   |   +--rw type?                identityref
|   |   +--rw id?                  uint16
|   |   +--rw source
|   |   |   +--rw id?        te-gen-node-id
|   |   |   +--rw type?      enumeration
|   +--rw global-source?             uint32
|   +--rw extended-id?               yang:hex-string
+--rw optimizations
|   +--rw (algorithm)?
|   |   +--:(metric) {path-optimization-metric}?
|   |   |   +--rw optimization-metric* [metric-type]
|   |   |   |   +--rw metric-type
|   |   |   |   |   identityref
|   |   |   |   +--rw weight?
|   |   |   |   |   uint8
|   |   |   +--rw explicit-route-exclude-objects
|   |   |   |   +--rw route-object-exclude-object*
|   |   |   |   |   [index]
|   |   |   |   |   +--rw index
|   |   |   |   |   |   uint32
|   |   |   |   +--rw (type)?
|   |   |   |   |   +--:(numbered-node-hop)
|   |   |   |   |   |   +--rw numbered-node-hop
|   |   |   |   |   |   |   +--rw node-id
|   |   |   |   |   |   |   |   te-node-id
|   |   |   |   |   |   |   +--rw hop-type?
|   |   |   |   |   |   |   |   te-hop-type
|   |   |   |   +--:(numbered-link-hop)

```

```

+--rw numbered-link-hop
+--rw link-tp-id
|   te-tp-id
+--rw hop-type?
|   te-hop-type
+--rw direction?
|   te-link-direction
+--:(unnumbered-link-hop)
+--rw unnumbered-link-hop
+--rw link-tp-id
|   te-tp-id
+--rw node-id
|   te-node-id
+--rw hop-type?
|   te-hop-type
+--rw direction?
|   te-link-direction
+--:(as-number)
+--rw as-number-hop
+--rw as-number
|   inet:as-number
+--rw hop-type?
|   te-hop-type
+--:(label)
+--rw label-hop
+--rw te-label
|   +--rw (technology)?
|   |   +--:(generic)
|   |   +--rw generic?
|   |   rt-types:ge
neralized-label
n
+--rw direction?
|   te-label-directio

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

```

```

| | | | | | | te-hop-type
| | | | | | | +---:(numbered-link-hop)
| | | | | | | | +--rw numbered-link-hop
| | | | | | | | | +--rw link-tp-id
| | | | | | | | | | te-tp-id
| | | | | | | | | +--rw hop-type?
| | | | | | | | | | te-hop-type
| | | | | | | | | +--rw direction?
| | | | | | | | | | te-link-direction
| | | | | | | +---:(unnumbered-link-hop)
| | | | | | | | +--rw unnumbered-link-hop
| | | | | | | | | +--rw link-tp-id
| | | | | | | | | | te-tp-id
| | | | | | | | | +--rw node-id
| | | | | | | | | | te-node-id
| | | | | | | | | +--rw hop-type?
| | | | | | | | | | te-hop-type
| | | | | | | | | +--rw direction?
| | | | | | | | | | te-link-direction
| | | | | | | +---:(as-number)
| | | | | | | | +--rw as-number-hop
| | | | | | | | | +--rw as-number
| | | | | | | | | | inet:as-number
| | | | | | | | | +--rw hop-type?
| | | | | | | | | | te-hop-type
| | | | | | | +---:(label)
| | | | | | | | +--rw label-hop
| | | | | | | | | +--rw te-label
| | | | | | | | | | +--rw (technology)?
| | | | | | | | | | | +---:(generic)
| | | | | | | | | | | | +--rw generic?
| | | | | | | | | | | | rt-types:ge
neralized-label
| | | | | | | | +--rw direction?
| | | | | | | | | te-label-directio
n
| | | | | | | +--rw tiebreakers
| | | | | | | | +--rw tiebreaker* [tiebreaker-type]
| | | | | | | | | +--rw tiebreaker-type identityref
+---:(objective-function)
| {path-optimization-objective-function}?
+--rw objective-function
| +--rw objective-function-type?
| identityref
+--rw named-path-constraint? leafref
| {te-types:named-path-constraints}?
+--rw te-bandwidth
| +--rw (technology)?

```

```

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

```

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


```

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

```

```

|                                     te-label-direction
|                                     +---rw label-step
|                                     |   +---rw (technology)?
|                                     |   +---:(generic)
|                                     |   +---rw generic?    int32
|                                     +---rw range-bitmap?   yang:hex-string
+---rw protection
|   +---rw enable?                                boolean
|   +---rw protection-type?                       identityref
|   +---rw protection-reversion-disable?          boolean
|   +---rw hold-off-time?                         uint32
|   +---rw wait-to-revert?                       uint16
|   +---rw aps-signal-id?                        uint8
+---rw restoration
|   +---rw enable?                                boolean
|   +---rw restoration-type?
|   |   identityref
|   +---rw restoration-scheme?
|   |   identityref
|   +---rw restoration-reversion-disable?         boolean
|   +---rw hold-off-time?                       uint32
|   +---rw wait-to-restore?                     uint16
|   +---rw wait-to-revert?                     uint16
+---ro computed-paths-properties
|   +---ro computed-path-properties* [k-index]
|   |   +---ro k-index                        uint8
|   |   +---ro path-properties
|   |   |   +---ro path-metric* [metric-type]
|   |   |   |   +---ro metric-type            identityref
|   |   |   |   +---ro accumulative-value?    uint64
|   |   |   +---ro path-affinities-values
|   |   |   |   +---ro path-affinities-value* [usage]
|   |   |   |   |   +---ro usage              identityref
|   |   |   |   |   +---ro value?            admin-groups
|   |   |   +---ro path-affinity-names
|   |   |   |   +---ro path-affinity-name* [usage]
|   |   |   |   |   +---ro usage              identityref
|   |   |   |   |   +---ro affinity-name* [name]
|   |   |   |   |   |   +---ro name          string
|   |   |   +---ro path-srlgs-lists
|   |   |   |   +---ro path-srlgs-list* [usage]
|   |   |   |   |   +---ro usage              identityref
|   |   |   |   |   +---ro values*          srlg
|   |   |   +---ro path-srlgs-names
|   |   |   |   +---ro path-srlgs-name* [usage]
|   |   |   |   |   +---ro usage              identityref
|   |   |   |   |   +---ro names*          string
|   |   +---ro path-route-objects

```

				+--ro path-route-object* [index]
				+--ro index
				uint32
				+--ro (type)?
				+--:(numbered-node-hop)
				+--ro numbered-node-hop
				+--ro node-id te-node-id
				+--ro hop-type?
				te-hop-type
				+--:(numbered-link-hop)
				+--ro numbered-link-hop
				+--ro link-tp-id te-tp-id
				+--ro hop-type?
				te-hop-type
				+--ro direction?
				te-link-direction
				+--:(unnumbered-link-hop)
				+--ro unnumbered-link-hop
				+--ro link-tp-id te-tp-id
				+--ro node-id
				te-node-id
				+--ro hop-type?
				te-hop-type
				+--ro direction?
				te-link-direction
				+--:(as-number)
				+--ro as-number-hop
				+--ro as-number
				inet:as-number
				+--ro hop-type?
				te-hop-type
				+--:(label)
				+--ro label-hop
				+--ro te-label
				+--ro (technology)?
				+--:(generic)
				+--ro generic?
				rt-types:gener
alized-label				+--ro direction?
				te-label-direction
				+--ro te-bandwidth
				+--ro (technology)?
				+--:(generic)
				+--ro generic? te-bandwidth
				+--ro disjointness-type?
				te-types:te-path-disjointness
				+--ro computed-path-error-infos

```

    +--ro computed-path-error-info* []
      +--ro error-description?  string
      +--ro error-timestamp?    yang:date-and-time
      +--ro error-reason?       identityref
+--ro lsp-provisioning-error-infos
  +--ro lsp-provisioning-error-info* []
    +--ro error-description?  string
    +--ro error-timestamp?    yang:date-and-time
    +--ro error-node-id?      te-types:te-node-id
    +--ro error-link-id?      te-types:te-tp-id
    +--ro lsp-id?             uint16
+--ro lsps
  +--ro lsp* [node lsp-id]
    +--ro tunnel-name?
      |      -> /te/lsps/lsp/tunnel-name
    +--ro node      -> /te/lsps/lsp/node
    +--ro lsp-id    -> /te/lsps/lsp/lsp-id
+--rw secondary-reverse-paths
  +--rw secondary-reverse-path* [name]
    +--rw name                                string
    +--rw path-computation-method?            identityref
    +--rw path-computation-server
      |   +--rw id?      te-gen-node-id
      |   +--rw type?    enumeration
    +--rw compute-only?                        empty
    +--rw use-path-computation?                boolean
    +--rw lockdown?                            empty
    +--ro path-scope?                          identityref
    +--rw preference?                          uint8
    +--rw association-objects
      +--rw association-object* [association-key]
        +--rw association-key    string
        +--rw type?              identityref
        +--rw id?                uint16
        +--rw source
          |   +--rw id?      te-gen-node-id
          |   +--rw type?    enumeration
      +--rw association-object-extended*
        [association-key]
        +--rw association-key    string
        +--rw type?              identityref
        +--rw id?                uint16
        +--rw source
          |   +--rw id?      te-gen-node-id
          |   +--rw type?    enumeration
        +--rw global-source?     uint32
        +--rw extended-id?       yang:hex-string
+--rw optimizations

```

```

+--rw (algorithm)?
+--:(metric) {path-optimization-metric}?
+--rw optimization-metric* [metric-type]
+--rw metric-type
|   identityref
+--rw weight?
|   uint8
+--rw explicit-route-exclude-objects
+--rw route-object-exclude-object*
|   [index]
+--rw index
|   uint32
+--rw (type)?
+--:(numbered-node-hop)
+--rw numbered-node-hop
+--rw node-id
|   te-node-id
+--rw hop-type?
|   te-hop-type
+--:(numbered-link-hop)
+--rw numbered-link-hop
+--rw link-tp-id
|   te-tp-id
+--rw hop-type?
|   te-hop-type
+--rw direction?
|   te-link-direction
+--:(unnumbered-link-hop)
+--rw unnumbered-link-hop
+--rw link-tp-id
|   te-tp-id
+--rw node-id
|   te-node-id
+--rw hop-type?
|   te-hop-type
+--rw direction?
|   te-link-direction
+--:(as-number)
+--rw as-number-hop
+--rw as-number
|   inet:as-number
+--rw hop-type?
|   te-hop-type
+--:(label)
+--rw label-hop
+--rw te-label
+--rw (technology)?
|   +--:(generic)

```

```

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

```

```

+---rw (technology)?
+---:(generic)
+---rw generic?
rt-types:ge

neralized-label

+---rw direction?
te-label-directio

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

```

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

bel

```

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

```

```

+--rw path-out-segment!
  +--rw label-restrictions
    +--rw label-restriction* [index]
      +--rw restriction?      enumeration
      +--rw index             uint32
      +--rw label-start
        +--rw te-label
          +--rw (technology)?
          |   +--:(generic)
          |   +--rw generic?
          |       rt-types:generalized-label
          +--rw direction?
          |       te-label-direction
          +--rw label-end
            +--rw te-label
              +--rw (technology)?
              |   +--:(generic)
              |   +--rw generic?
              |       rt-types:generalized-label
              +--rw direction?
              |       te-label-direction
              +--rw label-step
                +--rw (technology)?
                |   +--:(generic)
                |   +--rw generic?      int32
                +--rw range-bitmap?     yang:hex-string
      +--rw protection
        +--rw enable?                  boolean
        +--rw protection-type?         identityref
        +--rw protection-reversion-disable? boolean
        +--rw hold-off-time?           uint32
        +--rw wait-to-revert?          uint16
        +--rw aps-signal-id?           uint8
      +--rw restoration
        +--rw enable?                  boolean
        +--rw restoration-type?
        |   identityref
        +--rw restoration-scheme?
        |   identityref
        +--rw restoration-reversion-disable? boolean
        +--rw hold-off-time?           uint32
        +--rw wait-to-restore?         uint16
        +--rw wait-to-revert?          uint16
+--ro computed-paths-properties
  +--ro computed-path-properties* [k-index]
    +--ro k-index                     uint8
    +--ro path-properties
      +--ro path-metric* [metric-type]

```

```

|   +---ro metric-type          identityref
|   +---ro accumulative-value?  uint64
+---ro path-affinities-values
|   +---ro path-affinities-value* [usage]
|   +---ro usage                identityref
|   +---ro value?               admin-groups
+---ro path-affinity-names
|   +---ro path-affinity-name* [usage]
|   +---ro usage                identityref
|   +---ro affinity-name* [name]
|   +---ro name                 string
+---ro path-srlgs-lists
|   +---ro path-srlgs-list* [usage]
|   +---ro usage                identityref
|   +---ro values*             srlg
+---ro path-srlgs-names
|   +---ro path-srlgs-name* [usage]
|   +---ro usage                identityref
|   +---ro names*              string
+---ro path-route-objects
|   +---ro path-route-object* [index]
|   +---ro index
|   |   uint32
|   +---ro (type)?
|   |   +---:(numbered-node-hop)
|   |   |   +---ro numbered-node-hop
|   |   |   |   +---ro node-id      te-node-id
|   |   |   |   +---ro hop-type?
|   |   |   |   |   te-hop-type
|   |   |   +---:(numbered-link-hop)
|   |   |   |   +---ro numbered-link-hop
|   |   |   |   |   +---ro link-tp-id  te-tp-id
|   |   |   |   |   +---ro hop-type?
|   |   |   |   |   |   te-hop-type
|   |   |   |   |   +---ro direction?
|   |   |   |   |   |   te-link-direction
|   |   |   +---:(unnumbered-link-hop)
|   |   |   |   +---ro unnumbered-link-hop
|   |   |   |   |   +---ro link-tp-id  te-tp-id
|   |   |   |   |   +---ro node-id
|   |   |   |   |   |   te-node-id
|   |   |   |   |   +---ro hop-type?
|   |   |   |   |   |   te-hop-type
|   |   |   |   |   +---ro direction?
|   |   |   |   |   |   te-link-direction
|   |   +---:(as-number)
|   |   |   +---ro as-number-hop
|   |   |   +---ro as-number

```

```

|                                     |               inet:as-number
|                                     |       +---ro hop-type?
|                                     |               te-hop-type
|                                     |   +---:(label)
|                                     |       +---ro label-hop
|                                     |           +---ro te-label
|                                     |               +---ro (technology)?
|                                     |                   +---:(generic)
|                                     |                       +---ro generic?
|                                     |                           rt-types:gener
alized-label
|                                     |
|                                     |       +---ro direction?
|                                     |                               te-label-direction
|                                     |   +---ro te-bandwidth
|                                     |       +---ro (technology)?
|                                     |           +---:(generic)
|                                     |               +---ro generic?    te-bandwidth
|                                     |   +---ro disjointness-type?
|                                     |               te-types:te-path-disjointness
+---ro computed-path-error-infos
|   +---ro computed-path-error-info* []
|       +---ro error-description?      string
|       +---ro error-timestamp?        yang:date-and-time
|       +---ro error-reason?            identityref
+---ro lsp-provisioning-error-infos
|   +---ro lsp-provisioning-error-info* []
|       +---ro error-description?      string
|       +---ro error-timestamp?        yang:date-and-time
|       +---ro error-node-id?          te-types:te-node-id
|       +---ro error-link-id?          te-types:te-tp-id
|       +---ro lsp-id?                  uint16
+---ro lsps
|   +---ro lsp* [node lsp-id]
|       +---ro tunnel-name?
|           |         -> /te/lsp/lsp/tunnel-name
|       +---ro node             -> /te/lsp/lsp/node
|       +---ro lsp-id            -> /te/lsp/lsp/lsp-id
+---x tunnel-action
|   +---w input
|       |     +----w action-type?      identityref
|   +---ro output
|       +---ro action-result?          identityref
+---x protection-external-commands
|   +---w input
|       +---w protection-external-command?
|           |     identityref
|   +---w protection-group-ingress-node-id?
|       |     te-types:te-node-id
```

```

|         +---w protection-group-egress-node-id?
|         |         te-types:te-node-id
|         +---w path-ref?                                path-ref
|         +---w traffic-type?
|         |         enumeration
|         +---w extra-traffic-tunnel-ref?                tunnel-ref
+---ro lsp
  +---ro lsp* [tunnel-name lsp-id node]
    +---ro tunnel-name                                string
    +---ro lsp-id                                    uint16
    +---ro node
      |         te-types:te-node-id
    +---ro source?
      |         te-types:te-node-id
    +---ro destination?
      |         te-types:te-node-id
    +---ro tunnel-id?                                uint16
    +---ro extended-tunnel-id?                        yang:dotted-quad
    +---ro operational-state?                          identityref
    +---ro signaling-type?                            identityref
    +---ro origin-type?                              enumeration
    +---ro lsp-resource-status?                        enumeration
    +---ro lockout-of-normal?                          boolean
    +---ro freeze?                                    boolean
    +---ro lsp-protection-role?                        enumeration
    +---ro lsp-protection-state?                       identityref
    +---ro protection-group-ingress-node-id?
      |         te-types:te-node-id
    +---ro protection-group-egress-node-id?
      |         te-types:te-node-id
    +---ro lsp-record-route-information
      +---ro lsp-record-route-information* [index]
        +---ro index                                uint32
        +---ro (type)?
          +---:(numbered-node-hop)
            +---ro numbered-node-hop
              +---ro node-id        te-node-id
              +---ro flags*         path-attribute-flags
          +---:(numbered-link-hop)
            +---ro numbered-link-hop
              +---ro link-tp-id     te-tp-id
              +---ro flags*         path-attribute-flags
          +---:(unnumbered-link-hop)
            +---ro unnumbered-link-hop
              +---ro link-tp-id     te-tp-id
              +---ro node-id?      te-node-id
              +---ro flags*         path-attribute-flags
          +---:(label)

```

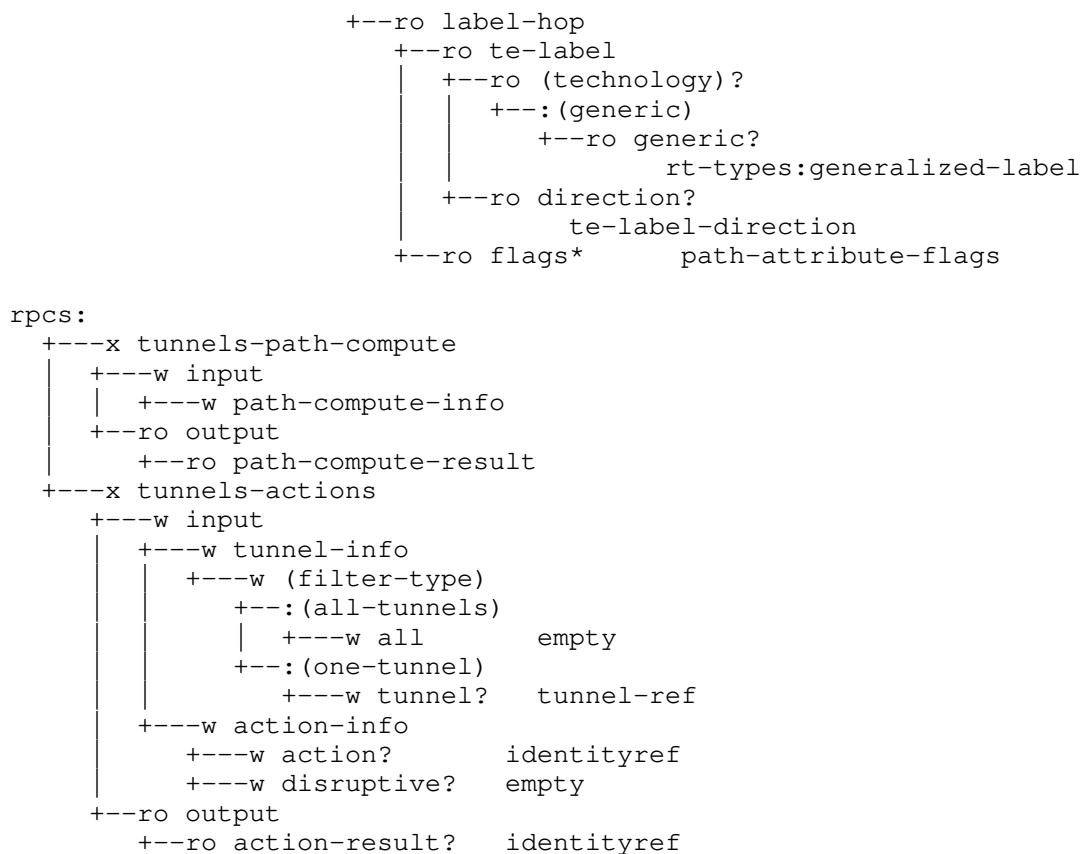


Figure 8: TE Tunnel generic model YANG tree diagram

5.3. YANG Module

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

- ```
* ietf-yang-types and ietf-inet-types defined in [RFC6991]
* ietf-te-types defined in [RFC8776]
```

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

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

 /* Replace with IANA when assigned */

 prefix te;

 /* Import TE generic types */

 import ietf-te-types {
 prefix te-types;
 reference
 "RFC8776: Common YANG Data Types for Traffic Engineering.";
 }
 import ietf-inet-types {
 prefix inet;
 reference
 "RFC6991: Common YANG Data Types.";
 }
 import ietf-yang-types {
 prefix yang;
 reference
 "RFC6991: Common YANG Data Types.";
 }
}

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

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

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

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

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

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

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

description
 "YANG data module for TE configuration, state, and RPCs.
 The model fully conforms to the Network Management
 Datastore Architecture (NMDA).

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

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

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

revision 2021-10-22 {
 description
 "Latest update to TE generic YANG module.";
 reference
 "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels
 and Interfaces.";
}

identity path-computation-error-reason {
 description
 "Base identity for path computation error reasons.";
}

identity path-computation-error-no-topology {
 base path-computation-error-reason;
 description
 "Path computation has failed because there is no topology
 with the provided topology-identifier.";
}

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



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

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

identity path-computation-error-no-inclusion-hop {
 base path-computation-error-reason;
 description
 "Path computation has failed because there is no
 node or link provided by one or more inclusion hops.";
 reference
 "RFC8685";
}

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

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

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

```
 reference
 "RFC8685";
 }

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

 identity path-computation-error-p2mp {
 base path-computation-error-reason;
 description
 "Path computation has failed because of P2MP reachability
 problem.";
 reference
 "RFC8306";
 }

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

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

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

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

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

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

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

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

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

```
identity tunnel-action-reoptimize {
 base tunnel-actions-type;
 description
 "Reoptimize tunnel action type.";
}

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

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

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

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

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

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

typedef tunnel-ref {
```

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

 typedef path-ref {
 type union {
 type leafref {
 path "/te:te/te:tunnels/te:tunnel/"
 + "te:primary-paths/te:primary-path/te:name";
 }
 type leafref {
 path "/te:te/te:tunnels/te:tunnel/"
 + "te:secondary-paths/te:secondary-path/te:name";
 }
 }
 description
 "This type is used by data models that need to reference
 configured primary or secondary path of a TE tunnel.";
 }

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

 /**
 * TE tunnel generic groupings
 */

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

```
 enum ip {
 description
 "IP address representation of the node identifier.";
 }
 enum dotted-quad {
 description
 "Dotted quad address representation of the node
 identifier.";
 }
 }
 description
 "Type of node identifier representation.";
}

grouping primary-path {
 description
 "The tunnel primary path properties.";
 uses path-common-properties;
 uses path-preference;
 uses k-requested-paths;
 uses path-compute-info;
 uses path-state;
}

grouping primary-reverse-path {
 description
 "The tunnel primary reverse path properties.";
 reference
 "RFC7551";
 uses path-common-properties;
 uses path-compute-info;
 uses path-state;
}

grouping secondary-path {
 description
 "The tunnel secondary path properties.";
 uses path-common-properties;
 uses path-preference;
 uses path-compute-info;
 uses protection-restoration-properties;
 uses path-state;
}

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

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

grouping path-common-properties {
 description
 "Common path attributes.";
 leaf name {
 type string;
 description
 "TE path name.";
 }
 leaf path-computation-method {
 type identityref {
 base te-types:path-computation-method;
 }
 default "te-types:path-locally-computed";
 description
 "The method used for computing the path, either
 locally computed, queried from a server or not
 computed at all (explicitly configured).";
 }
 container path-computation-server {
 when "derived-from-or-self(..path-computation-method, "
 + "'te-types:path-externally-queried') " {
 description
 "The path-computation server when the path is
 externally queried.";
 }
 uses te-generic-node-id;
 description
 "Address of the external path computation
 server.";
 }
 leaf compute-only {
 type empty;
 description
 "When set, the path is computed and updated whenever
 the topology is updated. No resources are committed
 or reserved in the network.";
 }
 leaf use-path-computation {
 when "derived-from-or-self(..path-computation-method, "
 + "'te-types:path-locally-computed') " {
 type boolean;
 }
 }
}
```

```
 default "true";
 description
 "When 'true' indicates the path is dynamically computed
 and/or validated against the Traffic-Engineering Database
 (TED), and when 'false' indicates no validation against
 the TED is required.";
}
leaf lockdown {
 type empty;
 description
 "Indicates no reoptimization to be attempted for this path.";
}
leaf path-scope {
 type identityref {
 base te-types:path-scope-type;
 }
 default "te-types:path-scope-end-to-end";
 config false;
 description
 "Path scope if segment or an end-to-end path.";
}
}

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

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

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

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



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

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

grouping k-requested-paths {
 description
 "The k-shortest paths requests.";
 leaf k-requested-paths {
 type uint8;
 default "1";
 description
 "The number of k-shortest-paths requested from the path
 computation server and returned sorted by its optimization
 objective. The value 0 all possible paths.";
 }
}

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

```
grouping path-state {
 description
 "TE per path state parameters.";
 uses path-computation-response;
 uses lsp-provisioning-error-info {
 augment "lsp-provisioning-error-infos/"
 + "lsp-provisioning-error-info" {
 description
 "Augmentation of LSP provisioning information under a
 specific path.";
 leaf lsp-id {
 type uint16;
 description
 "The LSP-ID for which path computation was performed.";
 }
 }
 }
}
container lsps {
 config false;
 description
 "The TE LSPs container.";
 list lsp {
 key "node lsp-id";
 description
 "List of LSPs associated with the tunnel.";
 leaf tunnel-name {
 type leafref {
 path "/te:te/te:lsps/te:lsp/te:tunnel-name";
 }
 description "TE tunnel name.";
 }
 leaf node {
 type leafref {
 path "/te:te/te:lsps/te:lsp/te:node";
 }
 description "The node where the LSP state resides on.";
 }
 leaf lsp-id {
 type leafref {
 path "/te:te/te:lsps/te:lsp/te:lsp-id";
 }
 description "The TE LSP identifier.";
 }
 }
}

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

```
grouping path-computation-response {
 description
 "Attributes reported by path computation response.";
 container computed-paths-properties {
 config false;
 description
 "Computed path properties container.";
 list computed-path-properties {
 key "k-index";
 description
 "List of computed paths.";
 leaf k-index {
 type uint8;
 description
 "The k-th path returned from the computation server.
 A lower k value path is more optimal than higher k
 value path(s)";
 }
 uses path-properties {
 description
 "The TE path computed properties.";
 }
 }
 }
}
container computed-path-error-infos {
 config false;
 description
 "Path computation information container.";
 list computed-path-error-info {
 description
 "List of path computation info entries.";
 leaf error-description {
 type string;
 description
 "Textual representation of the error occurred during
 path computation.";
 }
 leaf error-timestamp {
 type yang:date-and-time;
 description
 "Timestamp of last path computation attempt.";
 }
 leaf error-reason {
 type identityref {
 base path-computation-error-reason;
 }
 description
 "Reason for the path computation error.";
 }
 }
}
```

```
 }
 }
}

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

grouping protection-restoration-properties-state {
 description
 "Protection parameters grouping.";
 leaf lockout-of-normal {
 type boolean;
 default "false";
 }
}
```

```
description
 "When set to 'True', it represents a lockout of normal
 traffic external command. When set to 'False', it
 represents a clear lockout of normal traffic external
 command. The lockout of normal traffic command applies
 to this Tunnel.";
reference
 "RFC4427";
}
leaf freeze {
 type boolean;
 default "false";
 description
 "When set to 'True', it represents a freeze external command.
 When set to 'False', it represents a clear freeze external
 command. The freeze command applies to all the Tunnels which
 are sharing the protection resources with this Tunnel.";
 reference
 "RFC4427";
}
leaf lsp-protection-role {
 type enumeration {
 enum working {
 description
 "A working LSP must be a primary LSP whilst a protecting
 LSP can be either a primary or a secondary LSP. Also,
 known as protected LSPs when working LSPs are associated
 with protecting LSPs.";
 }
 enum protecting {
 description
 "A secondary LSP is an LSP that has been provisioned
 in the control plane only; e.g. resource allocation
 has not been committed at the data plane.";
 }
 }
 default "working";
 description
 "LSP role type.";
 reference
 "RFC4872, section 4.2.1";
}
leaf lsp-protection-state {
 type identityref {
 base te-types:lsp-protection-state;
 }
 default "te-types:normal";
 description
```

```
 "The state of the APS state machine controlling which
 tunnels is using the resources of the protecting LSP.";
 }
 leaf protection-group-ingress-node-id {
 type te-types:te-node-id;
 default "0.0.0.0";
 description
 "Indicates the te-node-id of the protection group
 ingress node when the APS state represents an external
 command (LoP, SF, MS) applied to it or a WTR timer
 running on it. If the external command is not applied to
 the ingress node or the WTR timer is not running on it,
 this attribute is not specified. A value 0.0.0.0 is used
 when the te-node-id of the protection group ingress node is
 unknown (e.g., because the ingress node is outside the scope
 of control of the server)";
 }
 leaf protection-group-egress-node-id {
 type te-types:te-node-id;
 default "0.0.0.0";
 description
 "Indicates the te-node-id of the protection group egress node
 when the APS state represents an external command (LoP, SF,
 MS) applied to it or a WTR timer running on it. If the
 external command is not applied to the ingress node or
 the WTR timer is not running on it, this attribute is not
 specified. A value 0.0.0.0 is used when the te-node-id of
 the protection group ingress node is unknown (e.g., because
 the ingress node is outside the scope of control of the
 server)";
 }
}

grouping protection-restoration-properties {
 description
 "Protection and restoration parameters.";
 container protection {
 description
 "Protection parameters.";
 leaf enable {
 type boolean;
 default "false";
 description
 "A flag to specify if LSP protection is enabled.";
 reference
 "RFC4427";
 }
 leaf protection-type {
```

```
 type identityref {
 base te-types:lsp-protection-type;
 }
 default "te-types:lsp-protection-unprotected";
 description
 "LSP protection type.";
 }
 leaf protection-reversion-disable {
 type boolean;
 default "false";
 description
 "Disable protection reversion to working path.";
 }
 leaf hold-off-time {
 type uint32;
 units "milli-seconds";
 default "0";
 description
 "The time between the declaration of an SF or SD condition
 and the initialization of the protection switching
 algorithm.";
 reference
 "RFC4427";
 }
 leaf wait-to-revert {
 type uint16;
 units "seconds";
 description
 "Time to wait before attempting LSP reversion.";
 reference
 "RFC4427";
 }
 leaf aps-signal-id {
 type uint8 {
 range "1..255";
 }
 default "1";
 description
 "The APS signal number used to reference the traffic of
 this tunnel. The default value for normal traffic is 1.
 The default value for extra-traffic is 255. If not
 specified, non-default values can be assigned by the
 server, if and only if, the server controls both
 endpoints.";
 reference
 "RFC4427";
 }
}
```

```
container restoration {
 description
 "Restoration parameters.";
 leaf enable {
 type boolean;
 default "false";
 description
 "A flag to specify if LSP restoration is enabled.";
 reference
 "RFC4427";
 }
 leaf restoration-type {
 type identityref {
 base te-types:lsp-restoration-type;
 }
 default "te-types:lsp-restoration-restore-any";
 description
 "LSP restoration type.";
 }
 leaf restoration-scheme {
 type identityref {
 base te-types:restoration-scheme-type;
 }
 default "te-types:restoration-scheme-preconfigured";
 description
 "LSP restoration scheme.";
 }
 leaf restoration-reversion-disable {
 type boolean;
 default "false";
 description
 "Disable restoration reversion to working path.";
 }
 leaf hold-off-time {
 type uint32;
 units "milli-seconds";
 description
 "The time between the declaration of an SF or SD condition
 and the initialization of the protection switching
 algorithm.";
 reference
 "RFC4427";
 }
 leaf wait-to-restore {
 type uint16;
 units "seconds";
 description
 "Time to wait before attempting LSP restoration.";
```



```
 reference
 "RFC4427";
 }
 leaf wait-to-revert {
 type uint16;
 units "seconds";
 description
 "Time to wait before attempting LSP reversion.";
 reference
 "RFC4427";
 }
 }
 }
}

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

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

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

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

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

grouping tunnel-common-attributes {
 description
 "Common grouping to define the TE tunnel parameters";
 leaf source {
 type te-types:te-node-id;
```

```
 description
 "TE tunnel source node ID.";
 }
 leaf destination {
 type te-types:te-node-id;
 description
 "TE tunnel destination node identifier.";
 }
 leaf src-tunnel-tp-id {
 type binary;
 description
 "TE tunnel source termination point identifier.";
 }
 leaf dst-tunnel-tp-id {
 type binary;
 description
 "TE tunnel destination termination point identifier.";
 }
 leaf bidirectional {
 type boolean;
 default "false";
 description
 "Indicates a bidirectional co-routed LSP.";
 }
}

grouping tunnel-hierarchy-properties {
 description
 "A grouping for TE tunnel hierarchy information.";
 container hierarchy {
 description
 "Container for TE hierarchy related information.";
 container dependency-tunnels {
 description
 "List of tunnels that this tunnel can be potentially
 dependent on.";
 list dependency-tunnel {
 key "name";
 description
 "A tunnel entry that this tunnel can potentially depend
 on.";
 leaf name {
 type leafref {
 path "/te:te/te:tunnels/te:tunnel/te:name";
 require-instance false;
 }
 description
 "Dependency tunnel name. The tunnel may not have been
```

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

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

```
 description
 "An alternate name of the TE tunnel that can be modified
 anytime during its lifetime.";
 }
 leaf identifier {
 type uint32;
 description
 "TE tunnel Identifier.";
 reference
 "RFC3209";
 }
 leaf color {
 type uint32;
 description "The color associated with the TE tunnel.";
 reference "RFC9012";
 }
 leaf description {
 type string;
 default "None";
 description
 "Textual description for this TE tunnel.";
 }
 leaf admin-state {
 type identityref {
 base te-types:tunnel-admin-state-type;
 }
 default "te-types:tunnel-admin-state-up";
 description
 "TE tunnel administrative state.";
 }
 leaf operational-state {
 type identityref {
 base te-types:tunnel-state-type;
 }
 config false;
 description
 "TE tunnel operational state.";
 }
 uses encoding-and-switching-type;
 uses tunnel-common-attributes;
 container controller {
 description
 "Contains tunnel data relevant to external controller(s).
 This target node may be augmented by external module(s),
 for example, to add data for PCEP initiated and/or
 delegated tunnels.";
 leaf protocol-origin {
 type identityref {
```

```
 base protocol-origin-type;
 }
 description
 "The protocol origin for instantiating the tunnel.";
}
leaf controller-entity-id {
 type string;
 description
 "An identifier unique within the scope of visibility that
 associated with the entity that controls the tunnel";
 reference "RFC8232";
}
}
leaf reoptimize-timer {
 type uint16;
 units "seconds";
 description
 "Frequency of reoptimization of a traffic engineered LSP.";
}
uses tunnel-associations-properties;
uses protection-restoration-properties;
uses te-types:tunnel-constraints;
uses tunnel-hierarchy-properties;
container primary-paths {
 description
 "The set of primary paths.";
 list primary-path {
 key "name";
 description
 "List of primary paths for this tunnel.";
 uses primary-path;
 container primary-reverse-path {
 description
 "The reverse primary path properties.";
 uses primary-reverse-path;
 container candidate-secondary-reverse-paths {
 description
 "The set of referenced candidate reverse secondary
 paths from the full set of secondary reverse paths
 which may be used for this primary path.";
 list candidate-secondary-reverse-path {
 key "secondary-path";
 ordered-by user;
 description
 "List of candidate secondary reverse path(s)";
 leaf secondary-path {
 type leafref {
 path "../.../.../.../.../..."
 }
 }
 }
 }
 }
 }
}
```

```
 + "te:secondary-reverse-paths/"
 + "te:secondary-reverse-path/te:name";
 }
 description
 "A reference to the secondary reverse path that
 should be utilised when the containing primary
 reverse path option is in use.";
 }
}
}
}
container candidate-secondary-paths {
 description
 "The set of candidate secondary paths which may be used
 for this primary path. When secondary paths are
 specified in the list the path of the secondary LSP in
 use must be restricted to those path options referenced.
 The priority of the secondary paths is specified within
 the list. Higher priority values are less preferred -
 that is to say that a path with priority 0 is the most
 preferred path. In the case that the list is empty, any
 secondary path option may be utilised when the current
 primary path is in use.";
 list candidate-secondary-path {
 key "secondary-path";
 ordered-by user;
 description
 "List of candidate secondary paths for this tunnel.";
 leaf secondary-path {
 type leafref {
 path "../../../../../te:secondary-paths/"
 + "te:secondary-path/te:name";
 }
 description
 "A reference to the secondary path that should be
 utilised when the containing primary path option is
 in use.";
 }
 leaf active {
 type boolean;
 config false;
 description
 "Indicates the current active path option that has
 been selected of the candidate secondary paths.";
 }
 }
}
}
```



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

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

```
}

grouping tunnel-protection-actions {
 description
 "Protection external command actions.";
 action protection-external-commands {
 input {
 leaf protection-external-command {
 type identityref {
 base te-types:protection-external-commands;
 }
 description
 "Protection external command.";
 }
 leaf protection-group-ingress-node-id {
 type te-types:te-node-id;
 description
 "When specified, indicates whether the action is
 applied on ingress node.
 By default, if neither ingress nor egress node-id
 is set, the action applies to ingress node only.";
 }
 leaf protection-group-egress-node-id {
 type te-types:te-node-id;
 description
 "When specified, indicates whether the action is
 applied on egress node.
 By default, if neither ingress nor egress node-id
 is set, the action applies to ingress node only.";
 }
 leaf path-ref {
 type path-ref;
 description
 "Indicates to which path the external command applies
 to.";
 }
 leaf traffic-type {
 type enumeration {
 enum normal-traffic {
 description
 "The manual-switch or forced-switch command applies
 to the normal traffic (this Tunnel).";
 }
 enum null-traffic {
 description
 "The manual-switch or forced-switch command applies
 to the null traffic.";
 }
 }
 }
 }
 }
}
```

```
 enum extra-traffic {
 description
 "The manual-switch or forced-switch command applies
 to the extra traffic (the extra-traffic Tunnel
 sharing protection bandwidth with this Tunnel).";
 }
 }
 description
 "Indicates whether the manual-switch or forced-switch
 commands applies to the normal traffic, the null traffic
 or the extra-traffic.";
 reference
 "RFC4427";
}
leaf extra-traffic-tunnel-ref {
 type tunnel-ref;
 description
 "In case there are multiple extra-traffic tunnels sharing
 protection bandwidth with this Tunnel (m:n protection),
 represents which extra-traffic Tunnel the manual-switch
 or forced-switch to extra-traffic command applies to.";
}
}
}

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

grouping lsp-record-route-information-state {
 description
 "LSP Recorded route information grouping.";
 container lsp-record-route-information {
 description
 "RSVP recorded route object information.";
 list lsp-record-route-information {
 when "../../origin-type = 'ingress'" {
 description
 "Applicable on ingress LSPs only.";
 }
 }
 key "index";
 description
 "Record route list entry.";
 uses te-types:record-route-state;
 }
}
```

```
 }

 grouping lsp-grouping {
 description
 "LSPs state operational data grouping.";
 container lsp {
 config false;
 description
 "TE LSPs state container.";
 list lsp {
 key "tunnel-name lsp-id node";
 unique "source destination tunnel-id lsp-id "
 + "extended-tunnel-id";
 description
 "List of LSPs associated with the tunnel.";
 leaf tunnel-name {
 type string;
 description "The TE tunnel name.";
 }
 leaf lsp-id {
 type uint16;
 description
 "Identifier used in the SENDER_TEMPLATE and the
 FILTER_SPEC that can be changed to allow a sender to
 share resources with itself.";
 reference
 "RFC3209";
 }
 leaf node {
 type te-types:te-node-id;
 description
 "The node where the TE LSP state resides on.";
 }
 uses lsp-properties-state;
 uses lsp-record-route-information-state;
 }
 }
 }

 /*** End of TE LSP groupings ***/
 /**
 * TE global generic groupings
 */
 /* Global named admin-groups configuration data */

 grouping named-admin-groups-properties {
 description
 "Global named administrative groups configuration
```

```
 grouping.";
 leaf name {
 type string;
 description
 "A string name that uniquely identifies a TE
 interface named admin-group.";
 }
 leaf bit-position {
 type uint32;
 description
 "Bit position representing the administrative group.";
 reference
 "RFC3209 and RFC7308";
 }
}

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

/* Global named admin-srlgs configuration data */

grouping named-srlgs {
 description
 "Global named SRLGs configuration grouping.";
 container named-srlgs {
 description
 "TE named SRLGs container.";
 list named-srlg {
 if-feature "te-types:named-srlg-groups";
 key "name";
 description
 "A list of named SRLG groups.";
 leaf name {
```

```
 type string;
 description
 "A string name that uniquely identifies a TE
 interface named SRLG.";
 }
 leaf value {
 type te-types:srlg;
 description
 "An SRLG value.";
 }
 leaf cost {
 type uint32;
 description
 "SRLG associated cost. Used during path to append
 the path cost when traversing a link with this SRLG.";
 }
}
}
}

/* Global named paths constraints configuration data */

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

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

grouping named-path-constraints {
 description
 "Global named path constraints configuration
 grouping.";
 container named-path-constraints {
 description
 "TE named path constraints container.";
 list named-path-constraint {
 if-feature "te-types:named-path-constraints";
 key "name";
 leaf name {
 type string;
 description
 "A string name that uniquely identifies a
 path constraint set.";
 }
 uses path-constraints-common;
 description
 "A list of named path constraints.";
 }
 }
}

/* TE globals container data */

grouping globals-grouping {
 description
```

```
 "Globals TE system-wide configuration data grouping.";
 container globals {
 description
 "Globals TE system-wide configuration data container.";
 uses named-admin-groups;
 uses named-srlgs;
 uses named-path-constraints;
 }
}

/* TE tunnels container data */

grouping tunnels-grouping {
 description
 "Tunnels TE configuration data grouping.";
 container tunnels {
 description
 "Tunnels TE configuration data container.";
 list tunnel {
 key "name";
 description
 "The list of TE tunnels.";
 uses tunnel-properties;
 uses tunnel-actions;
 uses tunnel-protection-actions;
 }
 }
}

/* TE LSPs ephemeral state container data */

grouping lsp-properties-state {
 description
 "LSPs state operational data grouping.";
 leaf source {
 type te-types:te-node-id;
 description
 "Tunnel sender address extracted from
 SENDER_TEMPLATE object.";
 reference
 "RFC3209";
 }
 leaf destination {
 type te-types:te-node-id;
 description
 "The tunnel endpoint address extracted from SESSION object.";
 reference
 "RFC3209";
 }
}
```



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

```
 switch in the path.";
 }
 leaf lsp-resource-status {
 type enumeration {
 enum primary {
 description
 "A primary LSP is a fully established LSP for which the
 resource allocation has been committed at the data
 plane.";
 }
 enum secondary {
 description
 "A secondary LSP is an LSP that has been provisioned
 in the control plane only; e.g. resource allocation
 has not been committed at the data plane.";
 }
 }
 default "primary";
 description
 "LSP resource allocation state.";
 reference
 "RFC4872, section 4.2.1";
 }
 uses protection-restoration-properties-state;
}

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

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

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

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

/* TE Tunnel RPCs/execution Data */

rpc tunnels-path-compute {
```

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

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

```

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

```

Figure 9: TE Tunnel data model YANG module

## 6. TE Device YANG Model

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

## 6.1. Module Structure

### 6.1.1. TE Interfaces

This branch of the model manages TE interfaces that are present on a device. Examples of TE interface properties are:

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

The derived state associated with interfaces is grouped under the interface "state" sub-container as shown in Figure 10. This covers state data such as:

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

```

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

```

Figure 10: TE interface state YANG subtree

## 6.2. Tree Diagram

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

```

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

```

```

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

```

```

+--ro upstream-info
 +--ro phop? te-types:te-tp-id
 +--ro neighbor
 | +--ro id? te-gen-node-id
 | +--ro type? enumeration
 +--ro label? rt-types:generalized-label

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

```

Figure 11: TE Tunnel device model YANG tree diagram

### 6.3. YANG Module

The device TE YANG module 'ietf-te-device' imports the following module(s):

- \* ietf-yang-types and ietf-inet-types defined in [RFC6991]
- \* ietf-interfaces defined in [RFC8343]
- \* ietf-routing-types defined in [RFC8294]
- \* ietf-te-types defined in [RFC8776]
- \* ietf-te defined in this document

```

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

 /* Replace with IANA when assigned */

 prefix te-dev;

 /* Import TE module */

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

```



```
 Engineering Tunnels and Interfaces";
}

/* Import TE types */

import ietf-te-types {
 prefix te-types;
 reference
 "RFC8776: Common YANG Data Types for Traffic Engineering.";
}
import ietf-interfaces {
 prefix if;
 reference
 "RFC8343: A YANG Data Model for Interface Management";
}
import ietf-routing-types {
 prefix rt-types;
 reference
 "RFC8294: Common YANG Data Types for the Routing Area";
}

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

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

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

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

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

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

 Editor: Igor Bryskin
 <mailto:i_bryskin@yahoo.com>";
description
 "YANG data module for TE device configurations,
 state, and RPCs. The model fully conforms to the
```

Network Management Datastore Architecture (NMDA).

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

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

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

```
revision 2021-10-22 {
 description
 "Latest update to TE device YANG module.";
 reference
 "RFCXXXX: A YANG Data Model for Traffic Engineering Tunnels
 and Interfaces";
}
```

```
/**
 * TE LSP device state grouping
 */
```

```
grouping lsps-device-info {
 description
 "TE LSP device state grouping.";
 container lsp-timers {
 when "../te:origin-type = 'ingress'" {
 description
 "Applicable to ingress LSPs only.";
 }
 description
 "Ingress LSP timers.";
 leaf life-time {
 type uint32;
 units "seconds";
 description
 "TE LSP lifetime.";
 }
 leaf time-to-install {
```

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

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

/**
 * Device general groupings.
 */

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

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

```
grouping interfaces-grouping {
 description
 "TE interface configuration data grouping.";
 container interfaces {
 description
 "Configuration data model for TE interfaces.";
 uses te-all-attributes;
 list interface {
 key "interface";
 description
 "TE interfaces.";
 leaf interface {
 type if:interface-ref;
 description
 "TE interface name.";
 }
 /* TE interface parameters */
 uses te-attributes;
 }
 }
}

/**
 * TE interface device groupings
 */

grouping te-admin-groups-config {
 description
 "TE interface affinities grouping.";
 choice admin-group-type {
 description
 "TE interface administrative groups
 representation type.";
 case value-admin-groups {
 choice value-admin-group-type {
 description
 "choice of admin-groups.";
 case admin-groups {
 description
 "Administrative group/Resource
 class/Color.";
 leaf admin-group {
 type te-types:admin-group;
 description
 "TE interface administrative group.";
 }
 }
 }
 case extended-admin-groups {
```

```
 if-feature "te-types:extended-admin-groups";
 description
 "Extended administrative group/Resource
 class/Color.";
 leaf extended-admin-group {
 type te-types:extended-admin-group;
 description
 "TE interface extended administrative group.";
 }
 }
}
}
case named-admin-groups {
 list named-admin-groups {
 if-feature "te-types:extended-admin-groups";
 if-feature "te-types:named-extended-admin-groups";
 key "named-admin-group";
 description
 "A list of named admin-group entries.";
 leaf named-admin-group {
 type leafref {
 path "../..../te:globals/"
 + "te:named-admin-groups/te:named-admin-group/"
 + "te:name";
 }
 description
 "A named admin-group entry.";
 }
 }
}
}
}

/* TE interface SRLGs */

grouping te-srlgs-config {
 description
 "TE interface SRLG grouping.";
 choice srlg-type {
 description
 "Choice of SRLG configuration.";
 case value-srlgs {
 list values {
 key "value";
 description
 "List of SRLG values that
 this link is part of.";
 leaf value {
```

```
 type uint32 {
 range "0..4294967295";
 }
 description
 "Value of the SRLG";
 }
}
}
case named-srlgs {
 list named-srlgs {
 if-feature "te-types:named-srlg-groups";
 key "named-srlg";
 description
 "A list of named SRLG entries.";
 leaf named-srlg {
 type leafref {
 path "../..../te:globals/"
 + "te:named-srlgs/te:named-srlg/te:name";
 }
 description
 "A named SRLG entry.";
 }
 }
}
}
}
}

grouping te-igp-flooding-bandwidth-config {
 description
 "Configurable items for igp flooding bandwidth
 threshold configuration.";
 leaf threshold-type {
 type enumeration {
 enum delta {
 description
 "'delta' indicates that the local
 system should flood IGP updates when a
 change in reserved bandwidth >= the specified
 delta occurs on the interface.";
 }
 enum threshold-crossed {
 description
 "THRESHOLD-CROSSED indicates that
 the local system should trigger an update (and
 hence flood) the reserved bandwidth when the
 reserved bandwidth changes such that it crosses,
 or becomes equal to one of the threshold values.";
 }
 }
 }
}
```

```
}
description
 "The type of threshold that should be used to specify the
 values at which bandwidth is flooded. 'delta' indicates that
 the local system should flood IGP updates when a change in
 reserved bandwidth >= the specified delta occurs on the
 interface. Where 'threshold-crossed' is specified, the local
 system should trigger an update (and hence flood) the
 reserved bandwidth when the reserved bandwidth changes such
 that it crosses, or becomes equal to one of the threshold
 values."
}
leaf delta-percentage {
 when "../threshold-type = 'delta'" {
 description
 "The percentage delta can only be specified when the
 threshold type is specified to be a percentage delta of
 the reserved bandwidth."
 }
 type rt-types:percentage;
 description
 "The percentage of the maximum-reservable-bandwidth
 considered as the delta that results in an IGP update
 being flooded."
}
leaf threshold-specification {
 when "../threshold-type = 'threshold-crossed'" {
 description
 "The selection of whether mirrored or separate threshold
 values are to be used requires user specified thresholds
 to be set."
 }
 type enumeration {
 enum mirrored-up-down {
 description
 "mirrored-up-down indicates that a single set of
 threshold values should be used for both increasing
 and decreasing bandwidth when determining whether
 to trigger updated bandwidth values to be flooded
 in the IGP TE extensions."
 }
 enum separate-up-down {
 description
 "separate-up-down indicates that a separate
 threshold values should be used for the increasing
 and decreasing bandwidth when determining whether
 to trigger updated bandwidth values to be flooded
 in the IGP TE extensions."
 }
 }
}
```



```
 }
 }
 description
 "This value specifies whether a single set of threshold
 values should be used for both increasing and decreasing
 bandwidth when determining whether to trigger updated
 bandwidth values to be flooded in the IGP TE extensions.
 'mirrored-up-down' indicates that a single value (or set of
 values) should be used for both increasing and decreasing
 values, where 'separate-up-down' specifies that the
 increasing and decreasing values will be separately
 specified."
 }
 leaf-list up-thresholds {
 when "../threshold-type = 'threshold-crossed'"
 + "and ../threshold-specification = 'separate-up-down'" {
 description
 "A list of up-thresholds can only be specified when the
 bandwidth update is triggered based on crossing a
 threshold and separate up and down thresholds are
 required."
 }
 type rt-types:percentage;
 description
 "The thresholds (expressed as a percentage of the maximum
 reservable bandwidth) at which bandwidth updates are to be
 triggered when the bandwidth is increasing."
 }
 }
 leaf-list down-thresholds {
 when "../threshold-type = 'threshold-crossed'"
 + "and ../threshold-specification = 'separate-up-down'" {
 description
 "A list of down-thresholds can only be specified when the
 bandwidth update is triggered based on crossing a
 threshold and separate up and down thresholds are
 required."
 }
 type rt-types:percentage;
 description
 "The thresholds (expressed as a percentage of the maximum
 reservable bandwidth) at which bandwidth updates are to be
 triggered when the bandwidth is decreasing."
 }
 }
 }
 leaf-list up-down-thresholds {
 when "../threshold-type = 'threshold-crossed'"
 + "and ../threshold-specification = 'mirrored-up-down'" {
 description
 "A list of thresholds corresponding to both increasing
```

```
 and decreasing bandwidths can be specified only when an
 update is triggered based on crossing a threshold, and
 the same up and down thresholds are required.";
 }
 type rt-types:percentage;
 description
 "The thresholds (expressed as a percentage of the maximum
 reservable bandwidth of the interface) at which bandwidth
 updates are flooded - used both when the bandwidth is
 increasing and decreasing.";
}
}

/* TE interface metric */

grouping te-metric-config {
 description
 "TE interface metric grouping.";
 leaf te-metric {
 type te-types:te-metric;
 description
 "TE interface metric.";
 }
}

/* TE interface switching capabilities */

grouping te-switching-cap-config {
 description
 "TE interface switching capabilities.";
 list switching-capabilities {
 key "switching-capability";
 description
 "List of interface capabilities for this interface.";
 leaf switching-capability {
 type identityref {
 base te-types:switching-capabilities;
 }
 description
 "Switching Capability for this interface.";
 }
 leaf encoding {
 type identityref {
 base te-types:lsp-encoding-types;
 }
 description
 "Encoding supported by this interface.";
 }
 }
}
```

```
 }
 }

 grouping te-advertisements-state {
 description
 "TE interface advertisements state grouping.";
 container te-advertisements-state {
 description
 "TE interface advertisements state container.";
 leaf flood-interval {
 type uint32;
 description
 "The periodic flooding interval.";
 }
 leaf last-flooded-time {
 type uint32;
 units "seconds";
 description
 "Time elapsed since last flooding in seconds.";
 }
 leaf next-flooded-time {
 type uint32;
 units "seconds";
 description
 "Time remained for next flooding in seconds.";
 }
 leaf last-flooded-trigger {
 type enumeration {
 enum link-up {
 description
 "Link-up flooding trigger.";
 }
 enum link-down {
 description
 "Link-down flooding trigger.";
 }
 enum threshold-up {
 description
 "Bandwidth reservation up threshold.";
 }
 enum threshold-down {
 description
 "Bandwidth reservation down threshold.";
 }
 enum bandwidth-change {
 description
 "Bandwidth capacity change.";
 }
 }
 }
 }
 }
}
```

```
 enum user-initiated {
 description
 "Initiated by user.";
 }
 enum srlg-change {
 description
 "SRLG property change.";
 }
 enum periodic-timer {
 description
 "Periodic timer expired.";
 }
 }
 default "periodic-timer";
 description
 "Trigger for the last flood.";
}
list advertised-level-areas {
 key "level-area";
 description
 "List of level-areas that the TE interface is advertised
 in.";
 leaf level-area {
 type uint32;
 description
 "The IGP area or level where the TE interface link state
 is advertised in.";
 }
}
}
}

/* TE interface attributes grouping */

grouping te-attributes {
 description
 "TE attributes configuration grouping.";
 uses te-metric-config;
 uses te-admin-groups-config;
 uses te-srlgs-config;
 uses te-igp-flooding-bandwidth-config;
 uses te-switching-cap-config;
 container state {
 config false;
 description
 "State parameters for interface TE metric.";
 uses te-advertisements-state;
 }
}
```

```
}

grouping te-all-attributes {
 description
 "TE attributes configuration grouping for all
 interfaces.";
 uses te-igp-flooding-bandwidth-config;
}

/** End of TE interfaces device groupings */
/**
 * TE device augmentations
 */

augment "/te:te" {
 description
 "TE global container.";
 /* TE Interface Configuration Data */
 uses interfaces-grouping;
 container performance-thresholds {
 description
 "Performance parameters configurable thresholds.";
 }
}

/* TE globals device augmentation */

augment "/te:te/te:globals" {
 description
 "Global TE device specific configuration parameters.";
 uses lsp-device-timers;
}

/* TE tunnels device configuration augmentation */

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

```

/* TE LSPs device state augmentation */

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

/* TE interfaces RPCs/execution Data */

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

<CODE ENDS>

```

Figure 12: TE device data model YANG module

## 7. Notifications

Notifications are a key component of any topology data model.

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

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

## 8. TE Generic and Helper YANG Modules

## 9. IANA Considerations

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

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

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

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

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

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

## 10. Security Considerations

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

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

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

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

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

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

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

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

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

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



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

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

## 11. Acknowledgement

The authors would like to thank the members of the multi-vendor YANG design team who are involved in the definition of this model.

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

## 12. Contributors

Himanshu Shah  
Ciena

Email: hshah@ciena.com

Xia Chen  
Huawei Technologies

Email: jescia.chenxia@huawei.com

Bin Wen  
Comcast

Email: Bin\_Wen@cable.comcast.com

## 13. Appendix A: Data Tree Examples

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

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

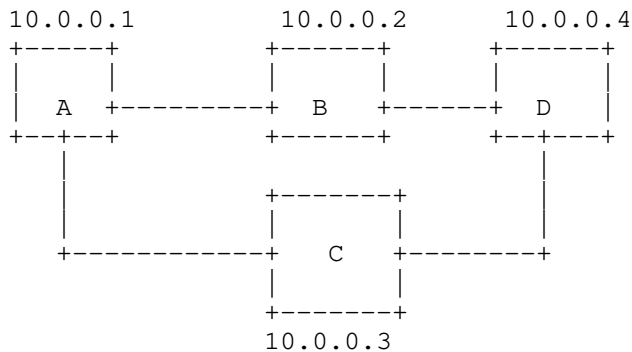


Figure 13: TE network used in data tree examples

### 13.1. Basic Tunnel Setup

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

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

```

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

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

```

### 13.2. Global Named Path Constraints

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

```
POST /restconf/data/ietf-te:te/globals/named-path-constraints HTTP/1.1
```

```
Host: example.com
```

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

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

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

### 13.3. Tunnel with Global Path Constraint

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

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

{
 "ietf-te:ietf-tunnel": [
 {
 "name": "Example_LSP_Tunnel_A_4_1",
 "encoding": "te-types:lsp-encoding-packet",
 "description": "Simple_LSP_with_named_path",
 "admin-state": "te-types:tunnel-state-up",
 "source": "10.0.0.1",
 "destination": "10.0.0.4",
 "signaling-type": "path-setup-rsvp",
 "bidirectional": "false",
 "primary-paths": [
 {
 "primary-path": {
 "name": "Simple_LSP_1",
 "use-path-computation": "true",
 "named-path-constraint": "max-hop-3"
 }
 }
]
 }
]
}
```

#### 13.4. Tunnel with Per-tunnel Path Constraint

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

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

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

### 13.5. Tunnel State

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

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

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

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

## 14. References

### 14.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <<https://www.rfc-editor.org/info/rfc3473>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC3945] Mannie, E., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, DOI 10.17487/RFC3945, October 2004, <<https://www.rfc-editor.org/info/rfc3945>>.
- [RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, DOI 10.17487/RFC4206, October 2005, <<https://www.rfc-editor.org/info/rfc4206>>.
- [RFC4872] Lang, J.P., Ed., Rekhter, Y., Ed., and D. Papadimitriou, Ed., "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery", RFC 4872, DOI 10.17487/RFC4872, May 2007, <<https://www.rfc-editor.org/info/rfc4872>>.
- [RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <<https://www.rfc-editor.org/info/rfc4875>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.

- [RFC6107] Shiomoto, K., Ed. and A. Farrel, Ed., "Procedures for Dynamically Signaled Hierarchical Label Switched Paths", RFC 6107, DOI 10.17487/RFC6107, February 2011, <<https://www.rfc-editor.org/info/rfc6107>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6780] Berger, L., Le Faucheur, F., and A. Narayanan, "RSVP ASSOCIATION Object Extensions", RFC 6780, DOI 10.17487/RFC6780, October 2012, <<https://www.rfc-editor.org/info/rfc6780>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", RFC 7308, DOI 10.17487/RFC7308, July 2014, <<https://www.rfc-editor.org/info/rfc7308>>.
- [RFC7551] Zhang, F., Ed., Jing, R., and R. Gandhi, Ed., "RSVP-TE Extensions for Associated Bidirectional Label Switched Paths (LSPs)", RFC 7551, DOI 10.17487/RFC7551, May 2015, <<https://www.rfc-editor.org/info/rfc7551>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.



- [RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <<https://www.rfc-editor.org/info/rfc8294>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <<https://www.rfc-editor.org/info/rfc8639>>.
- [RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <<https://www.rfc-editor.org/info/rfc8641>>.
- [RFC8776] Saad, T., Gandhi, R., Liu, X., Beeram, V., and I. Bryskin, "Common YANG Data Types for Traffic Engineering", RFC 8776, DOI 10.17487/RFC8776, June 2020, <<https://www.rfc-editor.org/info/rfc8776>>.
- [RFC8795] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Gonzalez de Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", RFC 8795, DOI 10.17487/RFC8795, August 2020, <<https://www.rfc-editor.org/info/rfc8795>>.

#### 14.2. Informative References

- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and  
P. Mattes, "Segment Routing Policy Architecture", Work in  
Progress, Internet-Draft, draft-ietf-spring-segment-  
routing-policy-16, 28 January 2022,  
<[https://www.ietf.org/archive/id/draft-ietf-spring-  
segment-routing-policy-16.txt](https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-16.txt)>.
- [I-D.ietf-teas-yang-rsvp]  
Beeram, V. P., Saad, T., Gandhi, R., Liu, X., and I.  
Bryskin, "A YANG Data Model for Resource Reservation  
Protocol (RSVP)", Work in Progress, Internet-Draft, draft-  
ietf-teas-yang-rsvp-17, 9 January 2022,  
<[https://www.ietf.org/archive/id/draft-ietf-teas-yang-  
rsvp-17.txt](https://www.ietf.org/archive/id/draft-ietf-teas-yang-rsvp-17.txt)>.
- [RFC4427] Mannie, E., Ed. and D. Papadimitriou, Ed., "Recovery  
(Protection and Restoration) Terminology for Generalized  
Multi-Protocol Label Switching (GMPLS)", RFC 4427,  
DOI 10.17487/RFC4427, March 2006,  
<<https://www.rfc-editor.org/info/rfc4427>>.
- [RFC8800] Litkowski, S., Sivabalan, S., Barth, C., and M. Negi,  
"Path Computation Element Communication Protocol (PCEP)  
Extension for Label Switched Path (LSP) Diversity  
Constraint Signaling", RFC 8800, DOI 10.17487/RFC8800,  
July 2020, <<https://www.rfc-editor.org/info/rfc8800>>.
- [RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder,  
"The BGP Tunnel Encapsulation Attribute", RFC 9012,  
DOI 10.17487/RFC9012, April 2021,  
<<https://www.rfc-editor.org/info/rfc9012>>.

#### Authors' Addresses

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

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

Xufeng Liu  
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram  
Juniper Networks

Email: vbeeram@juniper.net

Igor Bryskin  
Individual

Email: i\_bryskin@yahoo.com

Oscar Gonzalez de Dios  
Telefonica

Email: oscar.gonzalezdedios@telefonica.com

TEAS WG

Internet Draft  
Intended status: Informational

Young Lee  
Dhruv Dhody  
Huawei

Daniele Ceccarelli  
Ericsson

Oscar Gonzalez de Dios  
Telefonica

Expires: September 2017

March 13, 2017

Abstraction and Control of TE Networks (ACTN) Abstraction Methods  
draft-lee-teas-actn-abstraction-01

#### 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 September 13, 2017.

#### Copyright Notice

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

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

## Abstract

Abstraction and Control of Traffic Engineering (TE) Networks (ACTN) refers to the set of virtual network operations needed to orchestrate, control and manage large-scale multi-domain TE networks, so as to facilitate network programmability, automation, and efficient resource sharing.

As the ACTN architecture considers abstraction as one of the important building blocks, this document describes a few alternatives methods of abstraction for both packet and optical networks. This is an important consideration since the choice of the abstraction method impacts protocol design and the information it carries.

## Table of Contents

|                                                                                                    |    |
|----------------------------------------------------------------------------------------------------|----|
| 1. Introduction.....                                                                               | 3  |
| 2. ACTN Architecture.....                                                                          | 4  |
| 3. Abstraction Factors and Methods.....                                                            | 5  |
| 3.1. No abstraction (native/white topology).....                                                   | 6  |
| 3.2. One Abstract Node (black topology).....                                                       | 7  |
| 3.3. Abstraction of TE tunnels for all pairs of border nodes (grey topology).....                  | 9  |
| 3.3.1. Grey topology type A: border nodes with a TE links between them in a full mesh fashion..... | 10 |
| 3.3.2. Grey topology Type B.....                                                                   | 11 |
| 3.4. How to build grey topology.....                                                               | 11 |

|                                                                                              |    |
|----------------------------------------------------------------------------------------------|----|
| 3.4.1. Automatic generation of abstract topology by<br>configuration.....                    | 11 |
| 3.4.2. On-demand generation of supplementary topology via path<br>compute request/reply..... | 12 |
| 4. Protocol/Data Model Requirements.....                                                     | 13 |
| 4.1. Packet Networks.....                                                                    | 13 |
| 4.2. OTN Networks.....                                                                       | 14 |
| 4.3. WSON Networks.....                                                                      | 14 |
| 5. Security.....                                                                             | 14 |
| 6. Acknowledgements.....                                                                     | 15 |
| 7. References.....                                                                           | 15 |
| 7.1. Informative References.....                                                             | 15 |
| 8. Contributors.....                                                                         | 15 |
| Authors' Addresses.....                                                                      | 16 |
| Appendix A:.....                                                                             | 16 |

## 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 transport network) to provide connectivity and virtual network services for customers of the TE network. The services provided can be tuned to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers. More details about ACTN can be found in Section 2.

Abstraction is defined in [RFC7926] as:

Abstraction is the process of applying policy to the available TE information within a domain, to produce selective information that represents the potential ability to connect across the domain. Thus, abstraction does not necessarily offer all possible connectivity options, but presents a general view of potential connectivity according to the policies that determine how the domain's administrator wants to allow the domain resources to be used.

Connectivity referred to this document is TE path through a series of connected domains as used in [RFC7926].

As the ACTN architecture considers abstraction as one of the important building blocks, this document discusses a few alternatives for the methods of abstraction for both packet and optical networks. This is an important consideration since the choice of the abstraction method impacts protocol design and the information it carries.

The purpose of this document is to find a common agreement on the factors and methods of abstraction. These abstraction factors and methods may in turn impact implementations and protocol design.

## 2. ACTN Architecture

This section provides a brief description of ACTN architecture. [ACTN-Frame] describes the architecture model for ACTN including the entities (Customer Network Controller (CNC), Multi-domain Service Coordinator (MDSC), and Physical Network Controller (PNC) and their interfaces.

Figure 1 depicts a high-level control and interface architecture for ACTN and is a reproduction of Figure 5 from [ACTN-Frame].

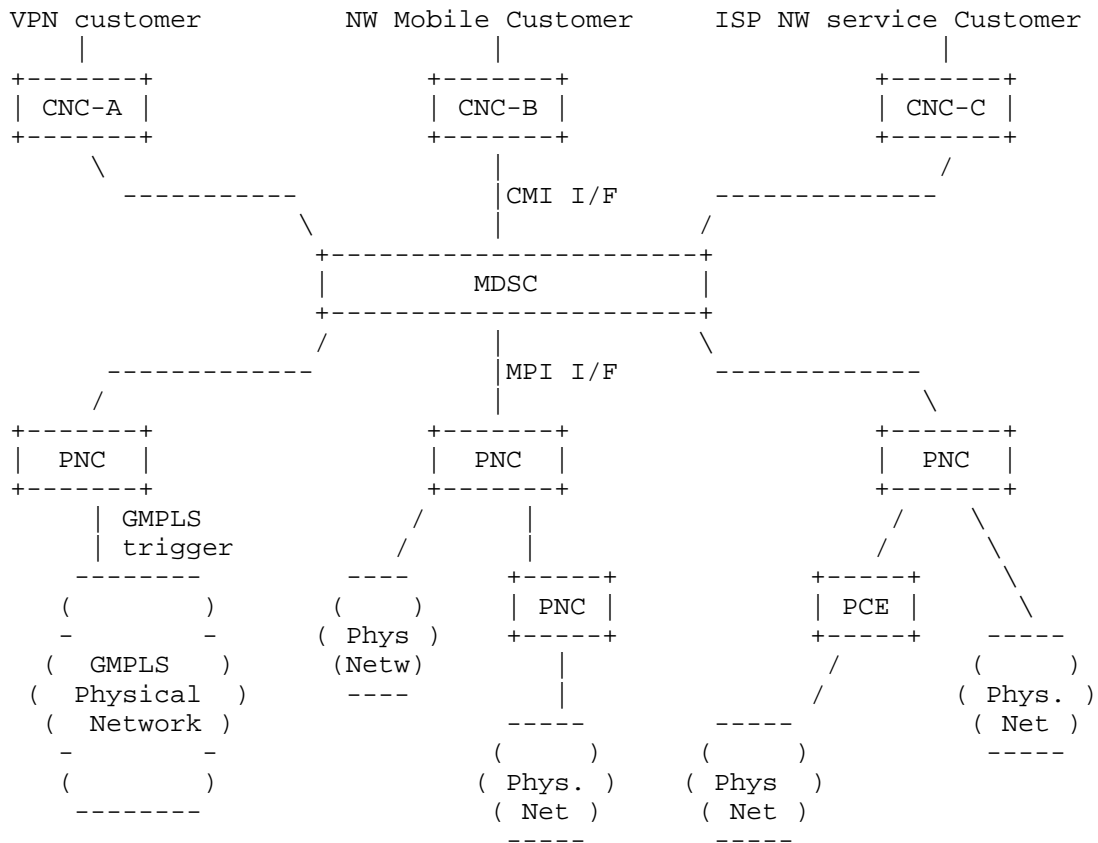


Figure 1 : ACTN Control Hierarchy

The MDSC oversees the specific aspects of the different domains and builds a single abstracted end-to-end network topology in order to coordinate end-to-end path computation and path/service provisioning. In order for the MDSC to perform its coordination function, it depends on the coordination with the PNCs which are the domain-level controllers especially as to what level of domain network resource abstraction is agreed upon between the MDSC and the PNCs.

As discussed in [RFC7926], abstraction is tied with policy of the networks. For instance, per an operational policy, the PNC would not be allowed to provide any technology specific details (e.g., optical parameters for WSON) in its update. In such case, the abstraction level of the update will be in a generic nature. In order for the MDSC to get technology specific topology information from the PNC, a request/reply mechanism may be employed.

In some cases, abstraction is also tied with the controller's capability of abstraction as abstraction involves some rules and algorithms to be applied to the actual network resource information (which is also known as network topology).

[TE-Topology] describes YANG models for TE-network abstraction. [PCEP-LS] describes PCEP Link-state mechanism that also allows for transport of abstract topology in the context of Hierarchical PCE.

### 3. Abstraction Factors and Methods

This section discusses factors that may impact the choice of abstraction and presents a number of abstraction methods.

It is important to understand that abstraction depends on several factors:

- The nature of underlying domain networks: Abstraction depends on the nature of the underlying domain networks. For instance, packet networks may have different level of abstraction requirements from that of optical networks. Within optical networks, WSON may have different level of abstraction requirements than the OTN networks.
- The capability of the PNC: Abstraction depends on the capability of the PNCs. As abstraction requires hiding details of the underlying resource network resource information, the PNC capability to run some internal optimization algorithm impacts the feasibility of abstraction. Some PNC may not have the ability to



abstract native topology while other PNCs may have such an ability to abstract actual topology by using sophisticated algorithms.

- Scalability factor: Abstraction is a function of scalability. If the actual network resource information is of small size, then the need for abstraction would be less than the case where the native network resource information is of large size. In some cases, abstraction may not be needed at all.
- The frequency of topology updates: The proper abstraction level may depend on the frequency of topology updates and vice versa.
- The capability/nature of the MDSC: The nature of the MDSC impacts the degree/level of abstraction. If the MDSC is not capable of handling optical parameters such as those specific to OTN/WSN, then white topology abstraction may not work well.
- The confidentiality: In some cases where the PNC would like to hide key internal topological data from the MDSC, the abstraction method should consider this aspect.
- The scope of abstraction: All of the aforementioned factors are equally applicable to both the MPI (MDSC-PNC Interface) and the CMI (CNC-MDSC Interface).

With having the aforementioned factors in mind, the following abstraction methods can be considered for implementations.

### 3.1. No abstraction (native/white topology)

This is a case where the PNC provides the actual network topology to the MDSC without any hiding or filtering. In this case, the MDSC has the full knowledge of the underlying network topology and as such there is no need for the MDSC to send a path computation request to the PNC. The computation burden will fall on the MDSC to find an optimal end-to-end path and optimal per domain paths.

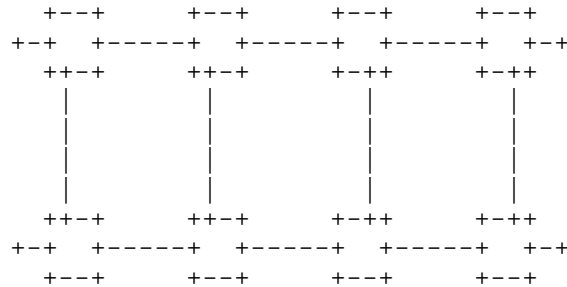


Figure 1: The native/white topology

### 3.2. One Virtual Node (black topology)

The entire domain network is abstracted as a single virtual node (see the definition of virtual node in [RFC7926]) with the access/egress links without disclosing any node internal connectivity information.

Figure 2a depicts a native topology with the corresponding black topology with one virtual node and inter-domain links. In this case, the MDSC has to make path computation requests to the PNCs before it can determine an end-to-end path. If there are a large number of inter-connected domains, this abstraction method may impose a heavy coordination load at the MDSC level in order to find an optimal end-to-end path.

Figure 2b depicts another type of a black topology with border nodes and inter-domain links.

The black topology would not give the MDSC any critical network resource information other than the border nodes/links information and as such it is likely to have a need for complementary communications between the MDSC and the PNCs (e.g., Path Computation Request/Reply).

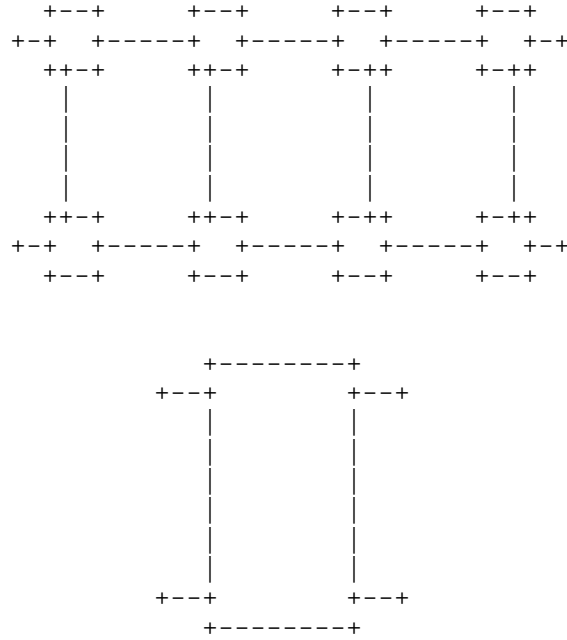


Figure 2a: The native topology and the corresponding black topology with one virtual node and inter-domain links

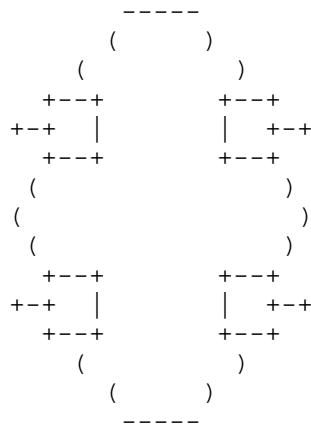


Figure 2b: A black topology with border nodes and inter-domain links

### 3.3. Abstraction of TE tunnels for all pairs of border nodes (grey topology)

This abstraction level, referred to a grey topology in [ACTN-frame] is between black topology and white topology from a granularity point of view. As shown in Figures 3a and 3b, we may further differentiate from a perspective of how to abstract internal TE resources between the pairs of border nodes:

- . Grey topology type A: border nodes with a TE links between them in a full mesh fashion (See Figure 3a)
- . Grey topology type B: border nodes with some internal abstracted nodes and abstracted links (See Figure 3b)

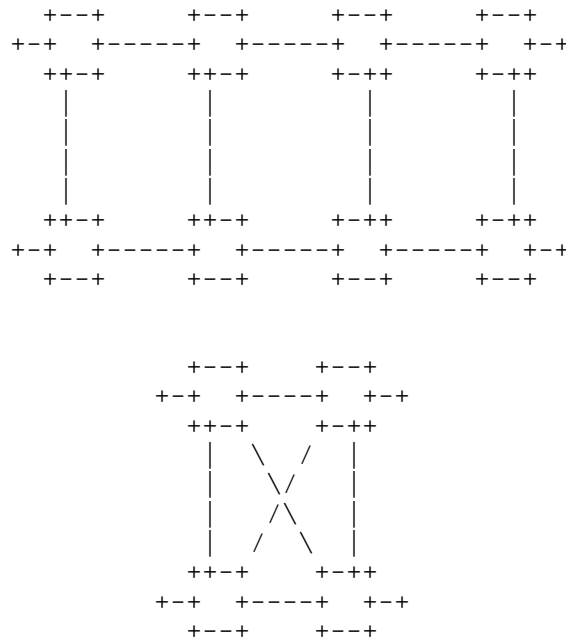


Figure 3a: The native topology and the corresponding grey topology type A with TE links between border nodes

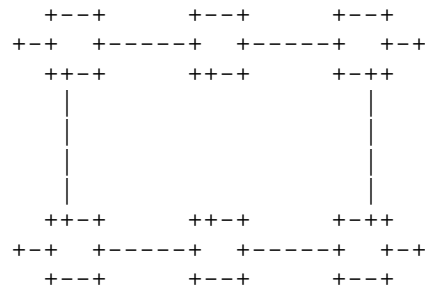


Figure 3b: The grey topology type B with abstract nodes/links between border nodes

### 3.3.1. Grey topology type A: border nodes with a TE links between them in a full mesh fashion

For each pair of ingress and egress nodes (i.e., border nodes to/from the domain), TE link metric is provided with TE attributes such as max bandwidth available, link delay, etc. This abstraction depends on the underlying TE networks.

Note that this topology is similar to the connectivity matrix defined in [TE-Topology]. The only thing might be different is some additional information about the end points of the links of the border nodes if they cannot be included in the connectivity matrix's termination points.

- For packet networks, abstraction may include max bandwidth available, delay, etc.
- For OTN networks, max bandwidth available may be per ODU 0/1/2/3 switching level or aggregated across all ODU switching levels (i.e., ODUj/k). Clearly, there is a trade-off between these two

abstraction methods. Some OTN switches can switch any level of ODUs and in such case there is no need for ODU level abstraction.

- For WSON networks, max bandwidth available may be per lambda/frequency level (OCh) or aggregated across all lambda/frequency level. Per OCh level abstraction gives more detailed data to the MDSC at the expense of more information processing. Either OCh-level or aggregated level abstraction should factor in the RWA constraint (i.e., wavelength continuity) at the PNC level. This means the PNC should have this capability and advertise it as such. See the Appendix for this abstraction method.

### 3.3.2. Grey topology Type B

The grey abstraction type B would allow the MDSC to have more information about the internals of the domain networks by the PNCs so that the MDSC can flexibly determine optimal paths. The MDSC may configure some of the internal virtual nodes (e.g., cross-connect) to redirect its traffic as it sees changes from the domain networks.

### 3.4. How to build grey topology

This section discusses two different methods of building a grey topology:

- . Automatic generation of abstract topology by configuration (Section 3.4.1)
- . On-demand generation of supplementary topology via path computation request/reply (Section 3.4.2)

#### 3.4.1. Automatic generation of abstract topology by configuration

The "Automatic generation" method is based on the abstraction/summarization of the whole domain by the PNC and its advertisement on MPI interface once the abstraction level is configured. The level of abstraction advertisement can be decided based on some PNC configuration parameters (e.g. provide the potential connectivity between any PE and any ASBR in an MPLS-TE network as described in section 3.3.1)

Note that the configuration parameters for this potential topology can include available B/W, latency, or any combination of defined parameters. How to generate such tunnel information is beyond the

scope of this document. Appendix A provides one example of this method for the WSON case.

Such potential topology needs to be periodically or incrementally/asynchronously updated every time that a failure, a recovery or the setup of new VNs causes a change in the characteristics of the advertised grey topology (e.g. in our previous case if due to changes in the network is it now possible to provide connectivity between a given PE and a given ASBR with a higher delay in the update).

### 3.4.2. On-demand generation of supplementary topology via path compute request/reply

The "on-demand generation" of supplementary topology is to be distinguished from automatic generation of abstract topology. While abstract topology is generated and updated automatically by configuration as explained in Section 3.4.1., additional supplementary topology may be obtained by the MDSC via path compute request/reply mechanism. Starting with a black topology advertisement from the PNCs, the MDSC may need additional information beyond the level of black topology from the PNCs. It is assumed that the black topology advertisement from PNCs would give the MDSC each domain's the border node/link information as described in Figure 2. Under this scenario, when the MDSC needs to allocate a new VN, the MDSC can issue a number of Path Computation requests as described in [ACTN-YANG] to different PNCs with constraints matching the VN request.

An example is provided in Figure 4, where the MDSC is requesting to setup a P2P VN between AP1 and AP2. The MDSC can use two different inter-domain links to get from Domain X to Domain Y, namely the one between ASBRX.1 and ASBRY.1 and the one between ASBRX.2 and ASBRY.2, but in order to choose the best end to end path it needs to know what domain X and Y can offer in term of connectivity and constraints between the PE nodes and the ASBR nodes.

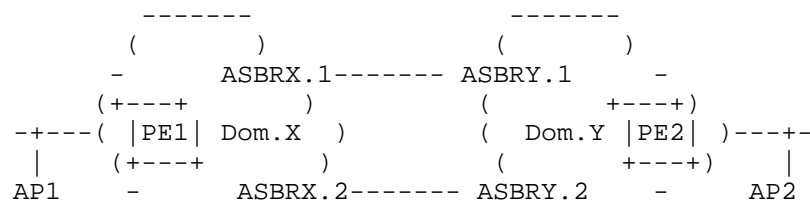




Figure 4: A multi-domain networks example

A path computation request will be issued to PNC.X asking for potential connectivity between PE1 and ASBRX.1 and between PE1 and ASBRX.2 with related objective functions and TE metric constraints. A similar request will be issued to PNC.Y and the results merged together at the MDSC to be able to compute the optimal end-to-end path including the inter domain links.

The info related to the potential connectivity may be cached by the MDSC for subsequent path computation processes or discarded, but in this case the PNCs are not requested to keep the grey topology updated.

#### 4. Protocol/Data Model Requirements

This section provides a set of requirements that may impact the way protocol/data model is designed and the information elements thereof which are carried in the protocol/data model.

It is expected that the abstraction level be negotiated between the CNC and the MDSC (i.e., the CMI) depending on the capability of the CNC. This negotiated level of abstraction on the CMI may also impact the way the MDSC and the PNCs configure and encode the abstracted topology. For example, if the CNC is capable of sophisticated technology specific operation, then this would impact the level of abstraction at the MDSC with the PNCs. On the other hand, if the CNC asks for a generic topology abstraction, then the level of abstraction at the MDSC with the PNCs can be less technology specific than the former case.

The subsequent sections provide a list of possible abstraction levels for various technology domain networks.

##### 4.1. Packet Networks

- For grey abstraction, the type of abstraction and its parameters MUST be defined and configured/negotiated.
  - o Abstraction Level 1: TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency



- o Other Level (TBD)

#### 4.2. OTN Networks

- For grey abstraction, the type of abstraction and its parameters MUST be defined and configured/negotiated.
  - o Abstraction Level 1: Per ODU Switching level (i.e., ODU type and number) TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Abstraction Level 2: Aggregated TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Other Level (TBD)

#### 4.3. WSON Networks

- For grey abstraction, the type of abstraction MUST and its parameters be defined and configured/negotiated.
  - o Abstraction Level 1: Per Lambda/Frequency level TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Abstraction Level 2: Aggregated TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Other Level (TBD)

Note that Appendix A provides how to compute WSON grey topology Abstraction Level 1 and Level 2. These examples illustrate that the encoding of an abstraction topology can be impacted by the configured/negotiated abstraction level in the ACTN interfaces.

## 5. Acknowledgements

We thank Name for providing useful comments and suggestions for this draft.

## 6. References

### 6.1. Informative References

- [RFC7926] A. Farrel, Ed., "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.
- [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.
- [PCEP-LS] D. Dhody, Y. Lee and D. Ceccarelli, "PCEP Extension for Distribution of Link-State and TE Information," draft-dhodylee-pce-pcep-ls, work in progress.
- [RFC7926] A. Farrel, et. al., "Problem Statement and Architecture for Information Exchange Between Interconnected Traffic Engineered Networks", RFC 7926, July 2016.
- [ACTN-YANG] X. Zhang, et. Al., "Applicability of YANG models for Abstraction and Control of Traffic Engineered Networks", draft-zhang-teas-actn-yang, work in progress

## 7. Contributors

### Contributor's Addresses

Sergio Belotti  
Nokia

Email: sergio.belotti@nokia.com

Xian Zhang  
Huawei

Email: zhang.xian@huawei.com

#### 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

Oscar Gonzalez de Dios  
Telefonica

Email: oscar.gonzalezdedios@telefonica.com

#### Appendix A:

This section provides how WSON grey topology abstraction levels 1 and 2 can be computed at a PNC. These examples illustrate that the encoding of an abstraction topology can be impacted by the configured/negotiated abstraction level at the MPI.

Abstraction Level 1: Per Lambda/Frequency level TE-tunnel abstraction for all (S-D) border pairs:

For each (S-D) border node pair,

- 1) The concept of a lambda plane: A lambda plane is a confined optical topology with respect to a given lambda value. If an OMS link has the wavelength of the given lambda available, it is included, otherwise excluded.
- 2) Calculate the maximal flow between S and D in every lambda plane. Max flow computation is restricted to each lambda plane is for OCh wavelength continuity.
- 3) Convert each feasible lambda plane with OCh wavelength continuity to B/W equivalent encoding; Send this per lambda level encoding for (S-D) to the MDSC;

Abstraction Level 2: Aggregated TE-tunnel abstraction for WSON for all (S-D) border pairs

For each (S-D) border node pair,

- 1) The concept of a lambda plane: A lambda plane is a confined optical topology with respect to a given lambda value. If an OMS link has the wavelength of the given lambda available, it is included, otherwise excluded.
- 2) Calculate the maximal flow between S and D in every lambda plane. Max flow computation is restricted to each lambda plane is for OCh wavelength continuity.
- 3) Add up the max flow values across all lambda planes. This is the maximal number of OCh paths that can be setup between S and D at the same time.
- 4) Convert the max number of OCh paths to B/W equivalent encoding; Send this encoding as max B/W for (S-D) to the MDSC;



TEAS WG

Internet Draft  
Intended status: Informational

Young Lee  
Dhruv Dhody  
Huawei

Daniele Ceccarelli  
Ericsson

Oscar Gonzalez de Dios  
Telefonica

Expires: December 2017

June 5, 2017

Abstraction and Control of TE Networks (ACTN) Abstraction Methods  
draft-lee-teas-actn-abstraction-02

#### 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 5, 2017.

#### 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.

## Abstract

Abstraction and Control of Traffic Engineering (TE) Networks (ACTN) refers to the set of virtual network operations needed to orchestrate, control and manage large-scale multi-domain TE networks, so as to facilitate network programmability, automation, and efficient resource sharing.

As the ACTN architecture considers abstraction as one of the important building blocks, this document describes a few alternatives methods of abstraction for both packet and optical networks. This is an important consideration since the choice of the abstraction method impacts protocol design and the information it carries.

## Table of Contents

|                                                                                            |    |
|--------------------------------------------------------------------------------------------|----|
| 1. Introduction.....                                                                       | 3  |
| 2. Abstraction Factors in ACTN Architecture.....                                           | 3  |
| 3. Build Method of Grey Topology.....                                                      | 6  |
| 3.1. Automatic generation of abstract topology by configuration                            | 6  |
| 3.2. On-demand generation of supplementary topology via path<br>compute request/reply..... | 6  |
| 4. Protocol/Data Model Requirements.....                                                   | 8  |
| 4.1. Packet Networks.....                                                                  | 8  |
| 4.2. OTN Networks.....                                                                     | 8  |
| 4.3. WSON Networks.....                                                                    | 9  |
| 5. Acknowledgements.....                                                                   | 11 |
| 6. References.....                                                                         | 11 |

|                                  |    |
|----------------------------------|----|
| 6.1. Informative References..... | 11 |
| 7. Contributors.....             | 11 |
| Authors' Addresses.....          | 12 |
| Appendix A:.....                 | 12 |

## 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 transport network) to provide connectivity and virtual network services for customers of the TE network. The services provided can be tuned to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers. More details about ACTN can be found in Section 2.

Abstraction is defined in [RFC7926] as:

Abstraction is the process of applying policy to the available TE information within a domain, to produce selective information that represents the potential ability to connect across the domain. Thus, abstraction does not necessarily offer all possible connectivity options, but presents a general view of potential connectivity according to the policies that determine how the domain's administrator wants to allow the domain resources to be used.

Connectivity referred to this document is TE path through a series of connected domains as used in [RFC7926].

As the ACTN architecture considers abstraction as one of the important building blocks, this document discusses a few alternatives for the methods of abstraction for both packet and optical networks. This is an important consideration since the choice of the abstraction method impacts protocol design and the information it carries.

The purpose of this document is to find a common agreement on the factors and methods of abstraction. These abstraction factors and methods may in turn impact implementations and protocol design.

## 2. Abstraction Factors in ACTN Architecture

This section provides abstraction factors in the ACTN architecture. [ACTN-Frame] describes the architecture model for ACTN including the entities (Customer Network Controller (CNC), Multi-domain Service



Coordinator (MDSC), and Physical Network Controller (PNC) and their interfaces.

The MDSC oversees the specific aspects of the different domains and builds a single abstracted end-to-end network topology in order to coordinate end-to-end path computation and path/service provisioning. In order for the MDSC to perform its coordination function, it depends on the coordination with the PNCs which are the domain-level controllers especially as to what level of domain network resource abstraction is agreed upon between the MDSC and the PNCs.

As discussed in [RFC7926], abstraction is tied with policy of the networks. For instance, per an operational policy, the PNC would not be allowed to provide any technology specific details (e.g., optical parameters for WSON) in its update. In such case, the abstraction level of the update will be in a generic nature. In order for the MDSC to get technology specific topology information from the PNC, a request/reply mechanism may be employed.

In some cases, abstraction is also tied with the controller's capability of abstraction as abstraction involves some rules and algorithms to be applied to the actual network resource information (which is also known as network topology).

[TE-Topology] describes YANG models for TE-network abstraction. [PCEP-LS] describes PCEP Link-state mechanism that also allows for transport of abstract topology in the context of Hierarchical PCE.

There are factors that may impact the choice of abstraction and presents a number of abstraction methods. It is important to understand that abstraction depends on several factors:

- The nature of underlying domain networks: Abstraction depends on the nature of the underlying domain networks. For instance, packet networks may have different level of abstraction requirements from that of optical networks. Within optical networks, WSON may have different level of abstraction requirements than the OTN networks.
- The capability of the PNC: Abstraction depends on the capability of the PNCs. As abstraction requires hiding details of the underlying resource network resource information, the PNC capability to run some internal optimization algorithm impacts the feasibility of abstraction. Some PNC may not have the ability to abstract native topology while other PNCs may have such an ability to abstract actual topology by using sophisticated algorithms.

- Scalability factor: Abstraction is a function of scalability. If the actual network resource information is of small size, then the need for abstraction would be less than the case where the native network resource information is of large size. In some cases, abstraction may not be needed at all.
- The frequency of topology updates: The proper abstraction level may depend on the frequency of topology updates and vice versa.
- The capability/nature of the MDSC: The nature of the MDSC impacts the degree/level of abstraction. If the MDSC is not capable of handling optical parameters such as those specific to OTN/WSN, then white topology abstraction may not work well.
- The confidentiality: In some cases where the PNC would like to hide key internal topological data from the MDSC, the abstraction method should consider this aspect.
- The scope of abstraction: All of the aforementioned factors are equally applicable to both the MPI (MDSC-PNC Interface) and the CMI (CNC-MDSC Interface).

[ACTN-Framework] defined the following three levels of topology abstraction and their descriptions:

- . White topology: this is a case where the PNC provides the actual network topology to the MDSC without any hiding or filtering.
- . Black topology: the entire domain network is abstracted as a single virtual node (see the definition of virtual node in [RFC7926]) with the access/egress links without disclosing any node internal connectivity information.
- . Grey topology: this abstraction level is between black topology and white topology from a granularity point of view. we may further differentiate from a perspective of how to abstract internal TE resources between the pairs of border nodes:
  - o Grey topology type A: border nodes with a TE links between them in a full mesh fashion
  - o Grey topology type B: border nodes with some internal abstracted nodes and abstracted links

### 3. Build Method of Grey Topology

This section discusses two different methods of building a grey topology:

- . Automatic generation of abstract topology by configuration (Section 3.1)
- . On-demand generation of supplementary topology via path computation request/reply (Section 3.2)

#### 3.1. Automatic generation of abstract topology by configuration

The "Automatic generation" method is based on the abstraction/summarization of the whole domain by the PNC and its advertisement on MPI interface once the abstraction level is configured. The level of abstraction advertisement can be decided based on some PNC configuration parameters (e.g. provide the potential connectivity between any PE and any ASBR in an MPLS-TE network as described in section 3.3.1)

Note that the configuration parameters for this potential topology can include available B/W, latency, or any combination of defined parameters. How to generate such tunnel information is beyond the scope of this document. Appendix A provides one example of this method for the WSON case.

Such potential topology needs to be periodically or incrementally/asynchronously updated every time that a failure, a recovery or the setup of new VNs causes a change in the characteristics of the advertised grey topology (e.g. in our previous case if due to changes in the network is it now possible to provide connectivity between a given PE and a given ASBR with a higher delay in the update).

#### 3.2. On-demand generation of supplementary topology via path compute request/reply

The "on-demand generation" of supplementary topology is to be distinguished from automatic generation of abstract topology. While abstract topology is generated and updated automatically by configuration as explained in Section 3.1., additional supplementary topology may be obtained by the MDSC via path compute request/reply mechanism. Starting with a black topology advertisement from the

PNCs, the MDSC may need additional information beyond the level of black topology from the PNCs. It is assumed that the black topology advertisement from PNCs would give the MDSC each domain's the border node/link information as described in Figure 2. Under this scenario, when the MDSC needs to allocate a new VN, the MDSC can issue a number of Path Computation requests as described in [ACTN-YANG] to different PNCs with constraints matching the VN request.

An example is provided in Figure 4, where the MDSC is requesting to setup a P2P VN between AP1 and AP2. The MDSC can use two different inter-domain links to get from Domain X to Domain Y, namely the one between ASBRX.1 and ASBRY.1 and the one between ASBRX.2 and ASBRY.2, but in order to choose the best end to end path it needs to know what domain X and Y can offer in term of connectivity and constraints between the PE nodes and the ASBR nodes.

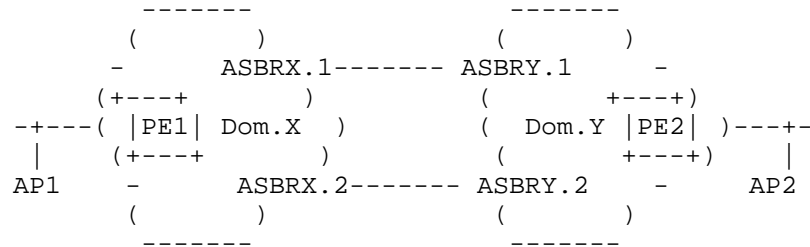


Figure 4: A multi-domain networks example

A path computation request will be issued to PNC.X asking for potential connectivity between PE1 and ASBRX.1 and between PE1 and ASBRX.2 with related objective functions and TE metric constraints. A similar request will be issued to PNC.Y and the results merged together at the MDSC to be able to compute the optimal end-to-end path including the inter domain links.

The info related to the potential connectivity may be cached by the MDSC for subsequent path computation processes or discarded, but in this case the PNCs are not requested to keep the grey topology updated.

#### 4. Protocol/Data Model Requirements

This section provides a set of requirements that may impact the way protocol/data model is designed and the information elements thereof which are carried in the protocol/data model.

It is expected that the abstraction level be negotiated between the CNC and the MDSC (i.e., the CMI) depending on the capability of the CNC. This negotiated level of abstraction on the CMI may also impact the way the MDSC and the PNCs configure and encode the abstracted topology. For example, if the CNC is capable of sophisticated technology specific operation, then this would impact the level of abstraction at the MDSC with the PNCs. On the other hand, if the CNC asks for a generic topology abstraction, then the level of abstraction at the MDSC with the PNCs can be less technology specific than the former case.

The subsequent sections provide a list of possible abstraction levels for various technology domain networks.

##### 4.1. Packet Networks

- For grey abstraction, the type of abstraction and its parameters MUST be defined and configured/negotiated.
  - o Abstraction Level 1: TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Other Level (TBD)

##### 4.2. OTN Networks

For OTN networks, max bandwidth available may be per ODU 0/1/2/3 switching level or aggregated across all ODU switching levels (i.e., ODUj/k). Clearly, there is a trade-off between these two abstraction methods. Some OTN switches can switch any level of ODUs and in such case there is no need for ODU level abstraction.

- For grey abstraction, the type of abstraction and its parameters MUST be defined and configured/negotiated.
  - o Abstraction Level 1: Per ODU Switching level (i.e., ODU type and number) TE-tunnel abstraction for all (S-D) border pairs with:

- . Maximum B/W available per Priority Level
  - . Minimum Latency
- o Abstraction Level 2: Aggregated TE-tunnel abstraction for all (S-D) border pairs with:
  - . Maximum B/W available per Priority Level
  - . Minimum Latency
- o Other Level (TBD)

#### 4.3. WSON Networks

For WSON networks, max bandwidth available may be per lambda/frequency level (OCh) or aggregated across all lambda/frequency level. Per OCh level abstraction gives more detailed data to the MDSC at the expense of more information processing. Either OCh-level or aggregated level abstraction should factor in the RWA constraint (i.e., wavelength continuity) at the PNC level. This means the PNC should have this capability and advertise it as such. See the Appendix for this abstraction method.

- For grey abstraction, the type of abstraction MUST and its parameters be defined and configured/negotiated.
  - o Abstraction Level 1: Per Lambda/Frequency level TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Abstraction Level 2: Aggregated TE-tunnel abstraction for all (S-D) border pairs with:
    - . Maximum B/W available per Priority Level
    - . Minimum Latency
  - o Other Level (TBD)

Examples: these examples show how to compute WSON grey topology Abstraction Level 1 and Level 2. These examples illustrate that the encoding of an abstraction topology can be impacted by the configured/negotiated abstraction level in the ACTN interfaces.

This section provides how WSON grey topology abstraction levels 1 and 2 can be computed at a PNC. These examples illustrate that the

encoding of an abstraction topology can be impacted by the configured/negotiated abstraction level at the MPI.

. Abstraction Level 1: Per Lambda/Frequency level TE-tunnel abstraction for all (S-D) border pairs:

For each (S-D) border node pair,

- 1) The concept of a lambda plane: A lambda plane is a confined optical topology with respect to a given lambda value. If an OMS link has the wavelength of the given lambda available, it is included, otherwise excluded.
- 2) Calculate the maximal flow between S and D in every lambda plane. Max flow computation is restricted to each lambda plane is for OCh wavelength continuity.
- 3) Convert each feasible lambda plane with OCh wavelength continuity to B/W equivalent encoding; Send this per lambda level encoding for (S-D) to the MDSC;

. Abstraction Level 2: Aggregated TE-tunnel abstraction for WSON for all (S-D) border pairs

For each (S-D) border node pair,

- 1) The concept of a lambda plane: A lambda plane is a confined optical topology with respect to a given lambda value. If an OMS link has the wavelength of the given lambda available, it is included, otherwise excluded.
- 2) Calculate the maximal flow between S and D in every lambda plane. Max flow computation is restricted to each lambda plane is for OCh wavelength continuity.
- 3) Add up the max flow values across all lambda planes. This is the maximal number of OCh paths that can be setup between S and D at the same time.
- 4) Convert the max number of OCh paths to B/W equivalent encoding; Send this encoding as max B/W for (S-D) to the MDSC;

## 5. Acknowledgements

We thank Adrian Farrel and Italo Busi for providing useful comments and suggestions for this draft.

## 6. References

### 6.1. Informative References

- [RFC7926] A. Farrel, Ed., "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.
- [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.
- [PCEP-LS] D. Dhody, Y. Lee and D. Ceccarelli, "PCEP Extension for Distribution of Link-State and TE Information," draft-dhodylee-pce-pcep-ls, work in progress.
- [RFC7926] A. Farrel, et. al., "Problem Statement and Architecture for Information Exchange Between Interconnected Traffic Engineered Networks", RFC 7926, July 2016.
- [ACTN-YANG] X. Zhang, et. Al., "Applicability of YANG models for Abstraction and Control of Traffic Engineered Networks", draft-zhang-teas-actn-yang, work in progress

## 7. Contributors

### Contributor's Addresses

Sergio Belotti  
Nokia

Email: sergio.belotti@nokia.com

Xian Zhang  
Huawei

Email: zhang.xian@huawei.com



## 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

Oscar Gonzalez de Dios  
Telefonica

Email: oscar.gonzalezdedios@telefonica.com



TEAS Working Group  
Internet Draft  
Intended Status: Standard Track

Y. Lee (Editor)  
Huawei  
D. Ceccarelli  
Ericsson  
Dhruv Doddy  
Huawei  
Takuya Miyasaka  
KDDI  
Peter Park  
KT  
Bin Young Yoon  
ETRI

Expires: January 5, 2017

July 5, 2016

A Yang Data Model for ACTN VN Operation

draft-lee-teas-actn-vn-yang-00.txt

#### Abstract

This document provides a YANG data model for the Abstraction and Control of Traffic Engineered (TE) networks (ACTN) Virtual Network (VN) 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 January 5, 2017.

Lee, et al.

Expires October 2016

[Page 1]

## Copyright Notice

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

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

## Table of Contents

|                                             |    |
|---------------------------------------------|----|
| 1. Introduction.....                        | 2  |
| 1.1. Terminology.....                       | 3  |
| 1.2. ACTN CMI context.....                  | 3  |
| 2. ACTN VN YANG Model (Tree Structure)..... | 7  |
| 3. ACTN-VN YANG Model.....                  | 8  |
| 4. Security Considerations.....             | 16 |
| 5. IANA Considerations.....                 | 16 |
| 6. Acknowledgments.....                     | 16 |
| 7. References.....                          | 17 |
| 7.1. Normative References.....              | 17 |
| 7.2. Informative References.....            | 17 |
| 8. Contributors.....                        | 17 |
| Authors' Addresses.....                     | 18 |

## 1. Introduction

This document provides a YANG data model for the Abstraction and Control of Traffic Engineered (TE) networks (ACTN) Virtual Network (VN) operation that is going to be implemented for the Customer Network Controller (CNC)- Multi-Domain Service Coordinator (MSDC) interface (CMI). The YANG model discussed in this document is used to operate customer-driven VNs during the VN instantiation and its life-cycle operations stages.

This document is based on the requirements identified in [ACTN-REQ] and on the architecture framework defined in [ACTN-FWK]. As defined in [ACTN-FW], a Virtual Network (VN) is a customer view of the TE networks and may comprise a set of end-to-end tunnels connecting customer's end points. Therefore, it is important to associate a VN with its members (which are a number of end-to-end tunnels) that are going to be created in the provider network. Each end-to-end tunnel defined under a VN is referred to as a VN member.

The YANG model discussed in this document basically provides the characteristics of VNs such as VN level parameters (e.g., VN ID, VN member, VN objective function, VN service preference, etc.), customer's end point characteristics (e.g., Customer Interface Capability, Access Points Interface characteristics, etc.), and other relevant VN information that needs to be known to the MDSC to facilitate ACTN VN operation.

### 1.1. Terminology

- Abstract Topology: Every lower controller in the provider network, when is representing its network topology to a higher layer, it may want to hide details of the actual network topology. In such case, an abstract topology may be used for this purpose. Abstract topology enhances scalability for the MDSC to operate multi-domain networks. The abstraction of topology can be applied on both MPI and CMI, from PNC to MDSC and from MDSC to CNC respectively.
- Access link: A link between a customer node and a provider node.
- Access Point (AP): An access point is defined on an access link. It is used to keep confidentiality between the customer and the provider. It is an identifier shared between the customer and the provider, used to map the end points of the border node in the provider NW. The AP can be used by the customer when requesting connectivity service to the provider. A number of parameters, e.g. available bandwidth, need to be associated to the AP to qualify it.
- VN Access Point (VNAP): A VNAP is defined within an AP as part of a given VN and is used to identify the portion of the AP, (and hence of the access link) dedicated to a given VN.

### 1.2. ACTN CMI context

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

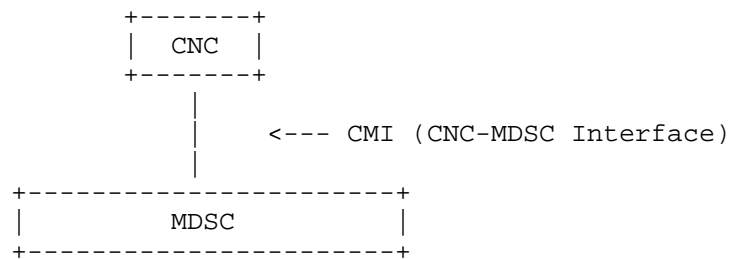


Figure 1. ACTN CMI

The CNC is the actor of the VN creation/modification/deletion (aka VN CRUD (- Create, read, update and delete) model). A VN comprises a set of the VN members. Each VN member is an end-to-end tunnel from a customer point of view that connects customer endpoints (i.e., source CE and destination CE). The following figure describes a VN that comprises three VN members forming a full mesh for the VN as an illustration.

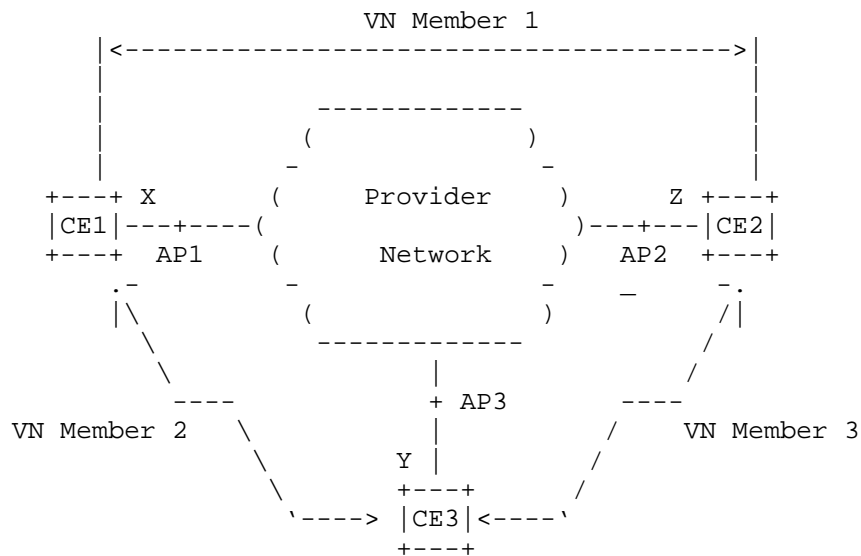


Figure 2. Full Mesh Example for a VN

In Figure 2, a VN has three members, namely, VN Member 1, VN member 2, and VN member 3. VN Member 1 is an end-to-end tunnel identified by CE1-AP1 (source) and CE2-AP2 (destination). Similarly, VN Member 2 by CE1-AP1 and CE3-AP3 and VN Member 3 by CE3-AP3 and CE2-AP2. This particular VN shown in Figure 2 is a full mesh connectivity across these three customer end-points.

It is also possible for the customer to create a VN which can be a hub and spoke or any other form of connectivity depending on its connectivity requirement. Each end-to-end tunnel may be

unidirectional or bidirectional which is also depending on its connectivity requirements. The following figure shows some examples of a VN that can be represented in a different connectivity form depending on the customer's connectivity requirements.

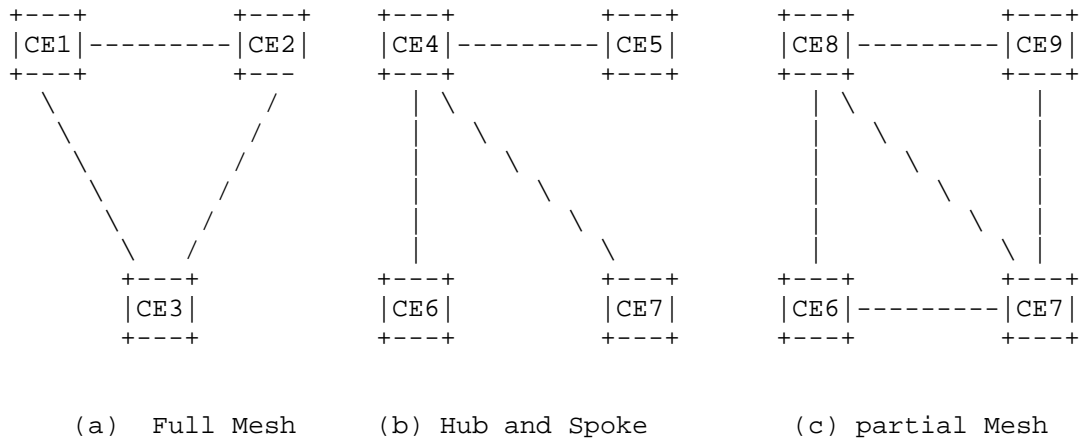


Figure 2. Different Connectivity Forms of a VN

It is important to note that a VN can associate a multiple number of end-to-end tunnels (i.e., VN members) with one unique identifier. From a customer standpoint, this simplifies its VN operation significantly.

The MDSC interacts with the CNC for the VN operation. Once the customer VN is requested by the CNC to the MDSC, the MDSC shall be responsible for translating and mapping the VN request into specific network centric-models (e.g., TE-tunnels [TE-Tunnel], TE-topology [TE-TOPO], etc.) to coordinate the multi-domain network operations with PNCs. The mapping and translation of a VN into network-centric models is out of the scope of this document.



The set of assumptions that applies to this document is the following:

- CNC is responsible for providing necessary Customer End-Points information to the MDSC via the CMI.
- The access links (between Customer Edge (CE) Devices and the Provider Edge (PE) Devices) are assumed to have been provisioned prior to the VN instantiation request.
- Access point identifiers have been configured and therefore are known in both the CNC and the MDSC.

## 2. 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? decimal64
 | | | +--rw avl-bandwidth? decimal64
 | +--rw vn
 | | +--rw vn-list* [vn-id]
 | | | +--rw vn-id uint32
 | | | +--rw vn-name? string
 | | | +--rw vn-member-list* [vn-member-id]
 | | | | +--rw vn-member-id uint32
 | | | | +--rw src? leafref
 | | | | +--rw src-vn-ap-id? uint32
 | | | | +--rw dest? leafref
 | | | | +--rw dest-vn-ap-id? uint32
 | | | +--rw delay? uint32
 | | | +--rw delay-variation? uint32
 | | | +--rw packet-loss? decimal64
 | | | +--rw bandwidth? decimal64
 | | | +--rw protection? identityref
 | | | +--rw local-reroute? boolean
 | | | +--rw push-allowed? boolean
 | | | +--rw incremental-update? boolean
 | | | +--rw admin-status? identityref
 | +--ro actn-state
 | | +--ro ap
 | | | +--ro access-point-list* [access-point-id]
 | | | | +--ro access-point-id uint32
 | | | | +--ro access-point-name? string
 | | | | +--ro max-bandwidth? decimal64

```

```

| +--ro avl-bandwidth? decimal64
+--ro vn
 +--ro vn-list* [vn-id]
 +--ro vn-id uint32
 +--ro vn-name? string
 +--ro vn-member-list* [vn-member-id]
 +--ro vn-member-id uint32
 +--ro src? leafref
 +--ro src-vn-ap-id? uint32
 +--ro dest? leafref
 +--ro dest-vn-ap-id? uint32
 +--ro delay? uint32
 +--ro delay-variation? uint32
 +--ro packet-loss? decimal64
 +--ro oper-status? identityref
 +--ro delay? uint32
 +--ro delay-variation? uint32
 +--ro packet-loss? decimal64
 +--ro bandwidth? decimal64
 +--ro protection? identityref
 +--ro local-reroute? boolean
 +--ro push-allowed? boolean
 +--ro incremental-update? boolean
 +--ro admin-status? identityref
 +--ro oper-status? Identityref

```

### 3. ACTN-VN YANG Code

The YANG code is as follows:

```

<CODE BEGINS> file ietf-actn-vn@2016-7-5.yang

module ietf-actn-vn {

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

 prefix "vn";

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

```

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

contact
 "Editor: Young Lee <leeyoung@huawei.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 creation
 /modification /deletion.";

revision 2016-07-05 {
 description
 "initial version.";
 reference
 "TBD";
}

/*
 * Groupings
 */

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 decimal64 {
```

```
 fraction-digits 2;
 range "0..max";
 }
 description
 "max bandwidth of the AP";
}
leaf avl-bandwidth {
 type decimal64 {
 fraction-digits 2;
 range "0..max";
 }
 description
 "available bandwidth of the AP";
}
/*add details and any other properties of AP,
not associated by a VN
CE port, PE port etc.

This link may not be in the TE topology model(?)
thus reference to that model would be incorrect
*/
} //access-point

grouping vn-member {
 description
 "vn-member is described by this container";
 leaf vn-member-id {
 type uint32;
 description
 "vn-member identifier";
 }
 leaf src {
 type leafref {
 path "/actn/ap/access-point-list/access-point-id";
 }
 description
 "reference to source AP";
 }
 leaf src-vn-ap-id {
 type uint32;
 description
```

```
 "vn-ap-id";
 }
 leaf dest {
 type leafref {
 path "/actn/ap/access-point-list/access-point-id";
 }
 description
 "reference to destination AP";
 }
 leaf dest-vn-ap-id {
 type uint32;
 description
 "vn-ap-id";
 }

 /* can we add reference to itef-te model(?) here
 */

} //vn-member

grouping connectivity-metric {
 description
 "service aware metrics";
 leaf delay {
 type uint32 {
 range "0..max";
 }
 description
 "Path Delay or latency in micro seconds.";
 }
 leaf delay-variation {
 type uint32 {
 range "0..max";
 }
 description
 "Path Delay variation in micro seconds.";
 }
 leaf packet-loss {
 type decimal64 {
 fraction-digits 6;
 range "0 .. max";
 }
 }
}
```

```
 }
 description
 "Path Packet Loss in percentage";
}
/*should we add other metrics
like bandwidth utilization?*/

} //connectivity-metric

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 objective-function {
 description
 "objective-function";

 uses connectivity-metric;

 leaf bandwidth {
 type decimal64 {
 fraction-digits 2;
 }
 }
}
```

```
 range "0..max";
 }
 description
 "bandwidth requested/required for
 vn-member-id";
}

leaf protection {
 type identityref {
 base te-types:lsp-prot-type;
 }
 description "protection type.";
}

uses policy;

} //objective-function

/*
 * 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";
 }
 list vn-member-list{
 key "vn-member-id";
 description
 "List of VN-members in a VN";
 uses vn-member;
 }
 uses objective-function;

 leaf admin-status {
 type identityref {
 base te-types:state-type;
 }
 default te-types:state-up;
 description "VN administrative state.";
 }
} //vn-list
} //vn
} //actn

/*
 * Operational data nodes
 */

container actn-state{
 config false;

 description
 "actn is described by this container";
```



```
 container ap {
 description
 "AP state";
 list access-point-list {
 key "access-point-id";
 description
 "access-point identifier";
 uses access-point {
 description
 "access-point information";
 }
 }
 }
 }
 container vn {
 description
 "VN state";
 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";
 }
 list vn-member-list {
 key "vn-member-id";
 description
 "List of VN-members in a VN";
 uses vn-member;
 uses connectivity-metric;
 leaf oper-status {
 type identityref {
 base te-types:state-type;
 }
 description

```

```
 "VN-member operational state.";
 }
}

uses objective-function;

leaf admin-status {
 type identityref {
 base te-types:state-type;
 }
 description "VN administrative state.";
}
leaf oper-status {
 type identityref {
 base te-types:state-type;
 }
 description "VN operational state.";
}
} //vn-list
} //vn
} //actn-state
}
```

<CODE ENDS>

#### 4. Security Considerations

TDB

#### 5. IANA Considerations

TDB

#### 6. Acknowledgments

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

## 7. References

### 7.1. Normative References

- [ACTN-REQ] Lee, et al., "Requirements for Abstraction and Control of TE Networks", work in progress: draft-ietf-teas-actn-requirements.
- [ACTN-FWK] D. Ceccarelli, Y. Lee [Editors], "Framework for Abstraction and Control of Traffic Engineered Networks", work in progress: draft-ceccarelli-teas-actn-framework.
- [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.

### 7.2. Informative References

## 8. 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

Authors' Addresses

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

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

Dhruv Dhoddy  
Huawei Technologies,  
Email: dhruv.ietf@gmail.com

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

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

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



TEAS Working Group  
Internet Draft  
Intended Status: Standard Track  
Expires: November 29, 2018

Y. Lee (Editor)  
Dhruv Dhody  
Huawei  
D. Ceccarelli  
Ericsson  
Igor Bryskin  
Huawei  
Bin Yeong Yoon  
ETRI

May 29, 2018

## A Yang Data Model for ACTN VN Operation

draft-lee-teas-actn-vn-yang-13

### 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 November 29, 2018.

Copyright Notice  
Lee, et al.

Expires November 2018

[Page 1]

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.....        | 6  |
| 3.1. Type 1 VN Illustration.....                      | 6  |
| 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.....                         | 10 |
| 4.2.1. VN Compute.....                                | 10 |
| 4.2.2. Multi-sources and Multi-destinations.....      | 11 |
| 4.2.3. Others.....                                    | 11 |
| 4.3. Summary.....                                     | 12 |
| 5. ACTN VN YANG Model (Tree Structure).....           | 12 |
| 6. ACTN-VN YANG Code.....                             | 15 |
| 7. JSON Example.....                                  | 27 |
| 7.1. ACTN VN JSON.....                                | 28 |
| 7.2. TE-topology JSON.....                            | 33 |
| 8. Security Considerations.....                       | 49 |
| 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 [Service-YANG]. The YANG model discussed in this document is used to operate customer-driven VNs during the VN computation, VN instantiation and its life-cycle 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 is performed with Connectivity Matrices sub-module of TE-Topology Model [TE-Topo] which interacts with the VN YANG module presented in this draft. 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] and [RFC7926] for the key terms used in this document.

### 2. ACTN CMI context

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

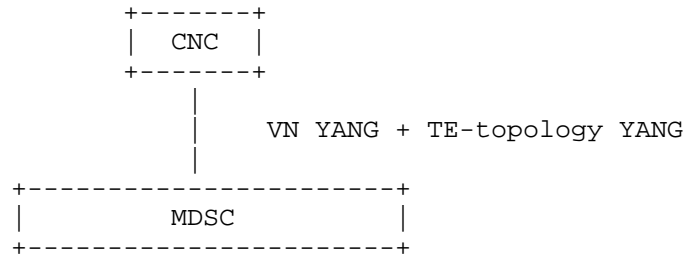


Figure 1. ACTN CMI

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 links (a Type 1 VN). Each 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:

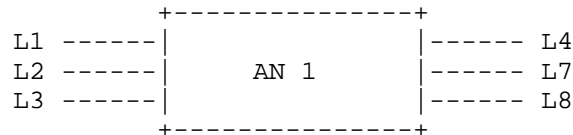


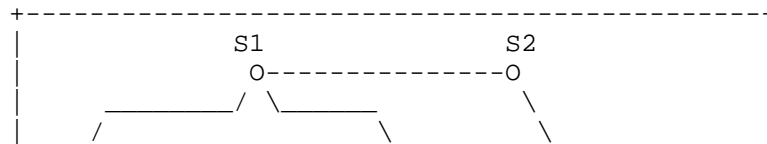
Figure 2. Abstract Node (One node topology)

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.



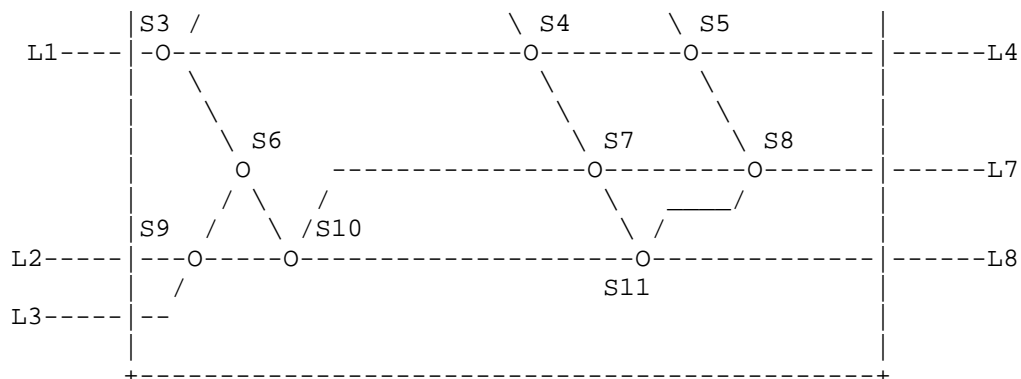


Figure 3. Type 2 topology

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

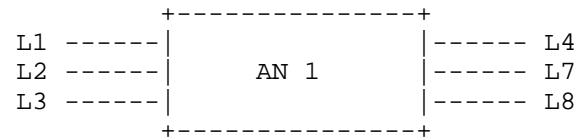
#### 3.1. Type 1 VN Illustration

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

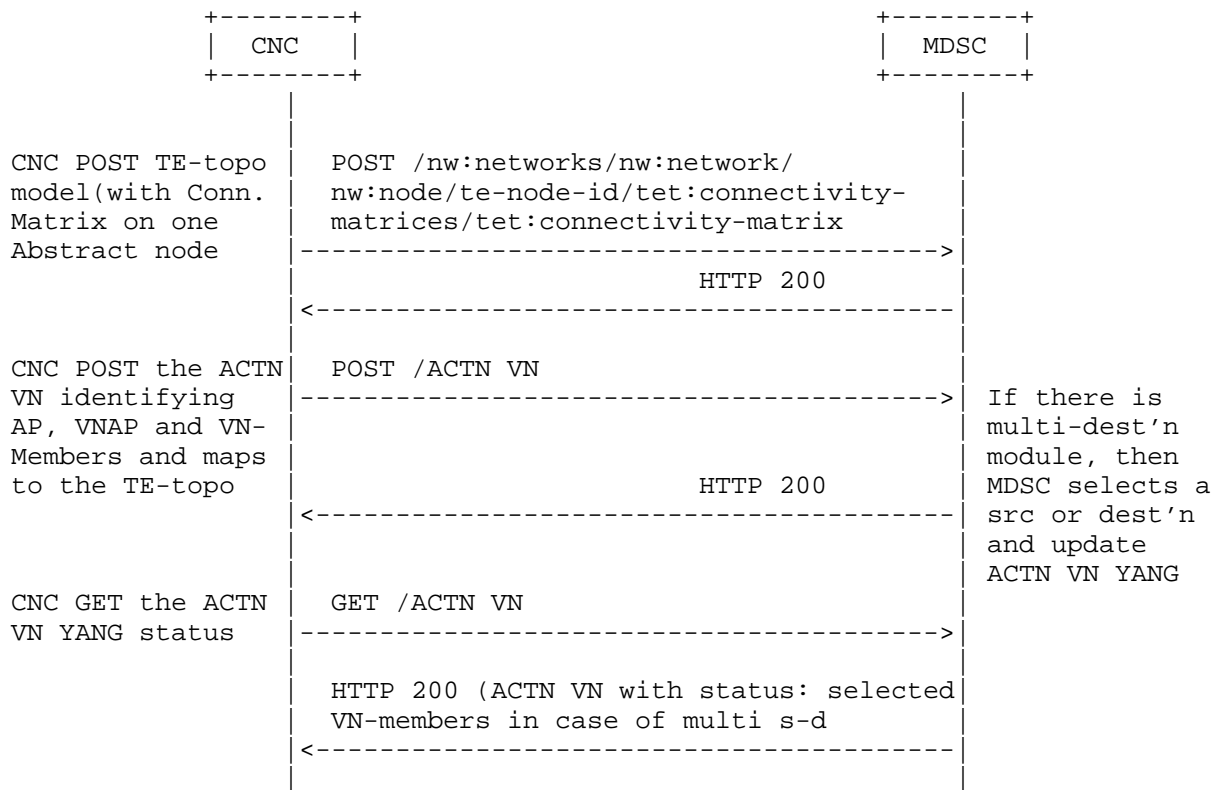
|             |       |
|-------------|-------|
| VN-Member 1 | L1-L4 |
| VN-Member 2 | L1-L7 |
| VN-Member 3 | L2-L4 |
| VN-Member 4 | L3-L8 |

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.

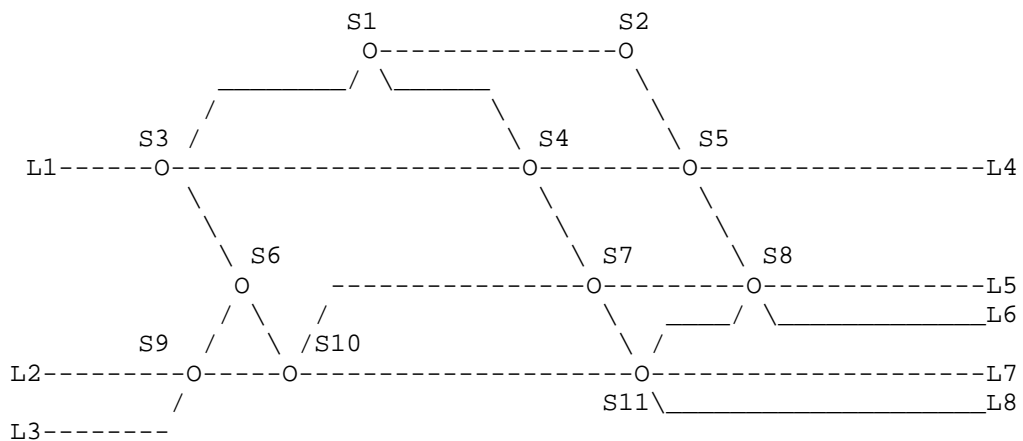


### 3.2. Type 2 VN Illustration

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)       |

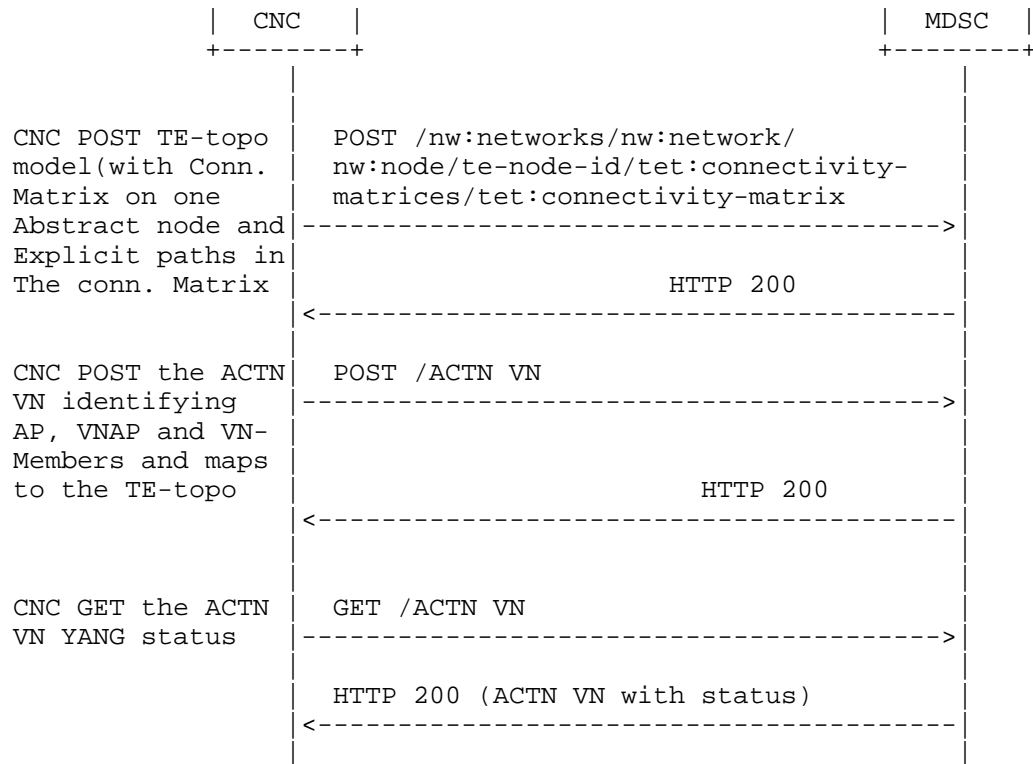
Where the following topology is the underlay for Abstraction Node 1 (AN1).



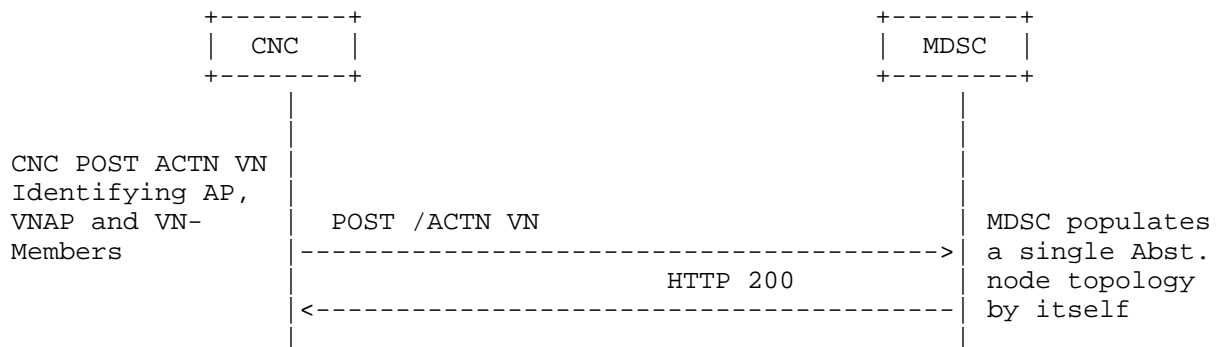
If CNC creates the single abstract topology, the following diagram shows the message flow between CNC and MDSC to instantiate this VN using ACTN VN and TE-Topology Model.

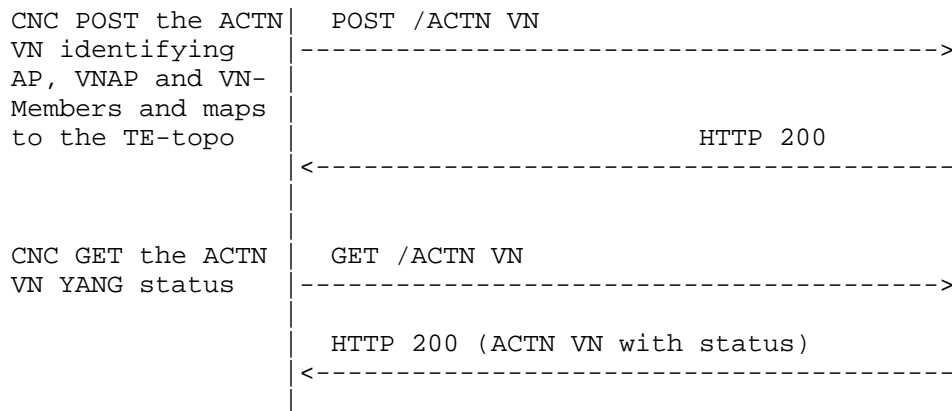
+-----+

+-----+



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 | | +--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-p
oint- | | +--rw dest-vn-ap-id? -> /actn/ap/access-point-list/vn-ap/vn
id | | +--rw multi-dest? boolean {multi-src-dest}?
 | | +--rw connectivity-matrix-id? -> /nw:networks/network/node/tet:
te/te- | +--ro oper-status? identityref
node-attributes/connectivity-matrices/connectivity-matrix/id
 | +--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? ->
| | +--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? ->
| | +--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? ->
| | +--rw 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 connetivity-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 connectivity-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 connectivity-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": "0x1p10"
 }
]
 }
 }
 }
 },
 "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.



## 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

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

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

Peter Park  
KT  
Email: peter.park@kt.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



Teas Working Group  
Internet Draft

Young Lee  
Huawei

Intended status: Informational

Sergio Belotti  
Nokia

Expires: April 2017

Dhruv Dhody  
Huawei

Daniele Ceccarelli  
Ericsson

Bin Young Yun  
ETRI

October 24, 2016

Information Model for Abstraction and Control of TE Networks (ACTN)

draft-leebelotti-teas-actn-info-05.txt

## Abstract

This draft provides an information model for Abstraction and Control of Traffic Engineered (TE) networks (ACTN).

## Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/ietf/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 April 24, 2015.

#### Copyright Notice

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

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

#### Table of Contents

|                                                    |    |
|----------------------------------------------------|----|
| 1. Introduction.....                               | 3  |
| 1.1. Terminology.....                              | 4  |
| 2. ACTN Common Interfaces Information Model.....   | 6  |
| 2.1. VN Action Primitives.....                     | 7  |
| 2.1.1. VN Instantiate.....                         | 7  |
| 2.1.2. VN Modify.....                              | 7  |
| 2.1.3. VN Delete.....                              | 8  |
| 2.1.4. VN Update.....                              | 8  |
| 2.1.5. VN Path Compute.....                        | 8  |
| 2.1.6. VN Query.....                               | 9  |
| 2.1.7. TE Update (for TE resources).....           | 9  |
| 2.2. VN Objects.....                               | 10 |
| 2.2.1. VN Identifier.....                          | 10 |
| 2.2.2. VN Service Characteristics.....             | 10 |
| 2.2.3. VN End-Point.....                           | 13 |
| 2.2.4. VN Objective Function.....                  | 13 |
| 2.2.5. VN Action Status.....                       | 14 |
| 2.2.6. VN Associated LSP.....                      | 14 |
| 2.2.7. VN Computed Path.....                       | 14 |
| 2.2.8. VN Service Preference.....                  | 15 |
| 2.3. Mapping of VN Primitives with VN Objects..... | 15 |
| 3. References.....                                 | 17 |

|                                                                                                            |    |
|------------------------------------------------------------------------------------------------------------|----|
| 3.1. Normative References.....                                                                             | 17 |
| 3.2. Informative References.....                                                                           | 17 |
| 4. Contributors.....                                                                                       | 18 |
| Contributors' Addresses.....                                                                               | 18 |
| Authors' Addresses.....                                                                                    | 18 |
| Appendix A: ACTN Applications.....                                                                         | 19 |
| A.1. Coordination of Multi-destination Service<br>Requirement/Policy.....                                  | 19 |
| A.2. Application Service Policy-aware Network Operation....                                                | 21 |
| A.3. Network Function Virtualization Service Enabled<br>Connectivity.....                                  | 23 |
| A.4. Dynamic Service Control Policy Enforcement for<br>Performance and Fault Management.....               | 25 |
| A.5. E2E VN Survivability and Multi-Layer (Packet-Optical)<br>Coordination for Protection/Restoration..... | 26 |

## 1. Introduction

This draft provides an information model for the requirements identified in the ACTN requirements [ACTN-Req] and the ACTN interfaces identified in the ACTN architecture and framework document [ACTN-Frame].

The purpose of this draft is to put all information elements of ACTN in one place before proceeding to development work necessary for protocol extensions and data models.

The ACTN reference architecture identified a three-tier control hierarchy as depicted in Figure 1:

- Customer Network Controllers (CNC)
- Multi-Domain Service Coordinator (MDSC)
- Physical Network Controllers (PNC).

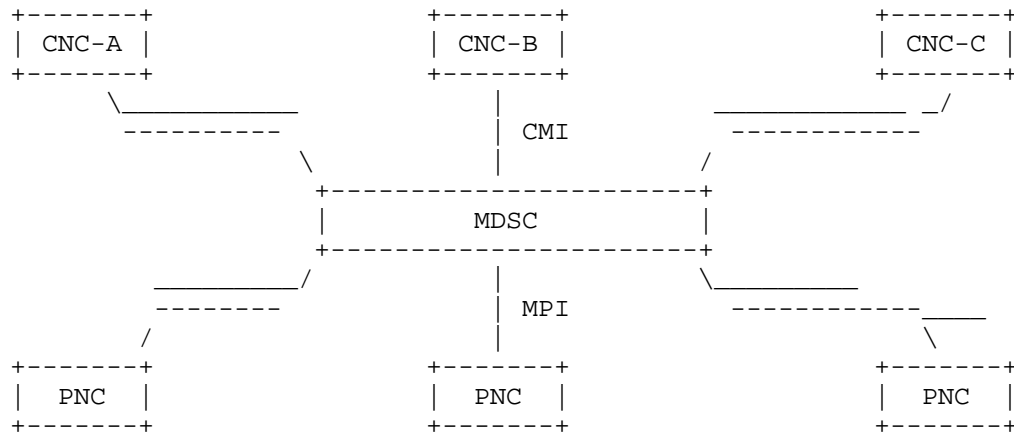


Figure 1: A Three-tier ACTN control hierarchy

The two interfaces with respect to the MDSC, one north of the MDSC and the other south of the MDSC are referred to as CMI (CNC-MDSC Interface) and MPI (MDSC-PNC Interface), respectively. It is intended to model these two interfaces and derivative interfaces thereof (e.g., MDSC to MSDC in a hierarchy of MDSCs) with one common model.

Appendix A provides some relevant ACTN use-cases extracted from [ACTN-Req]. Appendix A is information only and may help readers understand the context of key use-cases addressed in [ACTN-Req].

### 1.1. Terminology

- o A Virtual Network is a client view (typically a network slice) of the transport network. It is presented by the provider as a set of physical and/or abstracted resources. Depending on the agreement between client and provider various VN operations and VN views are possible. There are three aspects related to VN:

- 1) VN Creation: VN could be pre-configured and created via static negotiation between customer and provider. In other cases, VN could also be created dynamically based

on the request from the customer with given SLA attributes which satisfy the customer's objectives.

- 2) Dynamic Operations: VN could be further modified and deleted based on customer request to request changes in the network resources reserved for the customer. The customer can further act upon the virtual network resources to perform E2E tunnel management (set-up/release/modify). These changes will incur subsequent LSP management on the operator's level.
  - 3) VN View: (a) VN can be seen as an (or set of) e2e tunnel(s) from a customer point of view where an e2e tunnel is referred as a VN member. Each VN member (i.e., e2e tunnel) can then be formed by recursive aggregation of lower level paths at a provider level. Such end to end tunnels may comprise of customer end points, access links, intra domain paths and inter-domain link. In this view VN is thus a list of VN members. (b) VN can also be seen as a terms of topology comprising of physical and abstracted nodes and links. The nodes in this case include physical customer end points, border nodes, and internal nodes as well as abstracted nodes. Similarly the links includes physical access, inter-domain and intra-domain links as well as abstracted links. The abstracted nodes and links in this view can be pre-negotiated or created dynamically.
- o A Virtual Network Service (VNS) is the creation and offering of a Virtual Network by a provider to a customer in accordance with SLA agreements reached between them (e.g., re satisfying the customer's objectives).
  - o Abstraction is the process of applying policy to the available TE information within a domain, to produce selective information that represents the potential ability to connect across the domain. Thus, abstraction does not necessarily offer all possible connectivity options, but it presents a general view of potential connectivity according to the policies that determine how the domain's administrator wants to allow the domain resources to be used [RFC7926].
  - o Abstract topology: Every lower controller in the provider network, when is representing its network topology to a higher layer, it may want to selective hide details of the actual network topology, as suggested for abstraction in [RFC7926]. In such case, an abstract topology may be used for this purpose.



Abstract topology enhances scalability for the MDSC to operate multi-domain networks.

## 2. ACTN Common Interfaces Information Model

This section provides ACTN common interface information model to describe in terms of primitives, objects, their properties (represented as attributes), their relationships, and the resources for the service applications needed in the ACTN context.

Basic primitives (messages) are required between the CNC-MDSC and MDSC-PNC controllers. These primitives can then be used to support different ACTN network control functions like network topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options, etc.

The standard interface is described between a client controller and a server controller. A client-server relationship is recursive between a CNC and a MDSC and between a MDSC and a PNC. In the CMI, the client is a CNC while the server is a MDSC. In the MPI, the client is a MDSC and the server is a PNC. There may also be MDSC-MDSC interface(s) that need to be supported. This may arise in a hierarchy of MDSCs in which workloads may need to be partitioned to multiple MDSCs.

Basic primitives (messages) are required between the CNC-MDSC and MDSC-PNC controllers. These primitives can then be used to support different ACTN network control functions like network topology request/query, VN service request, path computation and connection control, VN service policy negotiation, enforcement, routing options, etc.

At a minimum, the following VN action primitives should be supported:

- VN Instantiate (See Section 2.1.1. for the description)
- VN Modify (See Section 2.1.2. for the description)
- VN Delete (See Section 2.1.3. for the description)
- VN Update ((See Section 2.1.4. for the description)
- VN Path Compute (See Section 2.1.5. for the description)

- VN Query (See Section 2.1.6. for the description)

In addition to VN action primitives, TE Update primitive should also be supported (See Section 2.1.7. for the description).

## 2.1. VN Action Primitives

This section provides a list of main primitives necessary to satisfy ACTN requirements specified in [ACTN-REQ].

<VN Action> describes main primitives. VN Action can be one of the following primitives: (i) VN Instantiate; (ii) VN Modify; (iii) VN Delete; (iv) VN Update; (v) VN Path Compute; (vi) VN Query.

```
<VN Action> ::= <VN Instantiate> |
 <VN Modify> |
 <VN Delete> |
 <VN Update> |
 <VN Path Compute> |
 <VN Query>
```

### 2.1.1. VN Instantiate

<VN Instantiate> refers to an action from customers/applications to request their VNs. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements. Please see the definition of VN in the section 2.

### 2.1.2. VN Modify

<VN Modify> refers to an action from customers/applications to modify an existing VN (i.e., instantiated VN). This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

### 2.1.3. VN Delete

<VN Delete> refers to an action from customers/applications to delete an existing VN. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

### 2.1.4. VN Update

<VN Update> refers to any update to the VN that need to be updated to the subscribers. VN Update fulfills a push model at CMI level, to make aware customers of any specific changes in the topology details related to VN instantiated.

Note the VN Update means the connection-related information (e.g., LSPs) update that has association with VNs.

### 2.1.5. VN Path Compute

<VN Path Compute> consists of Request and Reply. Request refers to an action from customers/applications to request a VN path computation. This primitive can also be applied from an MDSC to a PNC requesting a VN (if the domain the PNC supports can instantiate the entire VN) or a part of VN elements.

<VN Path Compute> Reply refers to the reply in response to <VN Path Compute> Request.

<VN Path Compute> Request/Reply is to be differentiated from a VN Instantiate. The purpose of VN Path Compute is a priori exploration to estimate network resources availability and getting a list of possible paths matching customer/applications constraints. To make this type of request Customer/application controller can have a shared (with lower controller) view of an abstract network topology on which to get the constraints used as input in a Path Computation request. The list of paths obtained by the request can be used by customer/applications to give path constraints during VNS connectivity request and to compel the lower level controller (e.g. MDSC) to select the path that Client/application controller has chosen among the set of paths returned by the Path Computation primitives. The importance of this primitives is for example in a scenario like multi-domain in which the optimal path obtained by an orchestrator as sum of optimal paths for different domain controller

cannot be the optimal path in the Client/application controller prospective. This only applies between CNC and MDSC.

#### 2.1.6. VN Query

<VN Query> refers to any query pertaining to the VN that has been already instantiated. VN Query fulfills a pull model and permit to get topology view.

<VN Query Reply> refers to the reply in response to <VN Query>.

#### 2.1.7. TE Update (for TE resources)

<TE Update> it is a primitives specifically related to MPI interface to provide TE resource update between any domain controller towards MDSC regarding the entire content of any "domain controller" TE topology or an abstracted filtered view of TE topology depending on negotiated policy.

<TE Update> ::= [<Abstraction>]<TE-topology...>

<TE-topology> ::= <TE-Topology-reference> <Node-list> <Link-list>

<Node-list> ::= <Node>[<Node-list>]

<Node> ::= <Node> <TE-Termination Points>

<Link-list> ::= <Link>[<Link-list>]

Where

<Abstraction> provides information on level of abstraction (as determined a priori).

<TE-topology-reference> ::= information related to the specific te-topology related to nodes and links present in this TE-topology.

<Node-list> ::= detailed information related to a specific node belonging to a te-topology e.g. te-node-attributes [TE-TOP0].

<Link-list> ::= information related to the specific link related belonging to a te-topology e.g. te-link-attributes [TE-TOP0].

<TE-Termination Points> ::= information details associated to the termination point of te-link related to a specific node e.g. interface-switching-capability [TE-TOP0].

## 2.2. VN Objects

This section provides a list of objects associated to VN action primitives.

### 2.2.1. VN Identifier

<VN Identifier> is a unique identifier of the VN.

### 2.2.2. VN Service Characteristics

VN Service Characteristics describes the customer/application requirements against the VNs to be instantiated.

<VN Service Characteristics> ::= <VN Connectivity Type>  
( <VN Traffic Matrix>... )  
<VN Survivability>

Where

<VN Connectivity Type> ::= <P2P> | <P2MP> | <MP2MP> | <MP2P> | <Multi-destination>

The Connectivity Type identifies the type of required VN Service. In addition to the classical type of services (e.g. P2P/P2MP etc.), ACTN defines the "multi-destination" service that is a new P2P service where the end points are not fixed. They can be chosen among a list of pre-configured end points or dynamically provided by the CNC.

<VN Traffic Matrix> ::= <Bandwidth>  
[ <VN Constraints> ]

The VN Traffic Matrix represents the traffic matrix parameters required against the service connectivity required and so the VN request instantiation between service related Access Points [ACTN-Frame]. Bandwidth is a mandatory parameter and a number of optional constrains can be specified in the <VN Constrains> (e.g. diversity,

cost). They can include objective functions and TE metrics bounds as specified in [RFC5441].

Further details on the VN constraints are specified below:

```
<VN Constraints> ::= [<Layer Protocol>]
 [<Diversity>]
 [<Shared Risk>]
 <Metric>
```

Where:

<Layer Protocol> Identifies the layer at which the VN service is requested. It could be for example MPLS, ODU, and OCh.

<Diversity> This allows asking for diversity constraints for a VN Instantiate/Modify or a VN Path Compute. For example, a new VN or a path is requested in total diversity from an existing one (e.g. diversity exclusion).

```
<Diversity> ::= <VN-exclusion> (<VN-id>...) |
 <VN-E2E Tunnel-exclusion> (<Tunnel-id>...)
```

<Shared Risk> Based on the realization of VN required, group of physical resources can be impacted by the same risk. An E2E tunnel can be impacted by this shared risk. This is used to get the SRLG associated with the different tunnels composing a VN.

<Metric> can include all the Metrics (cost, delay, delay variation, latency), bandwidth utilization parameters defined and referenced by [RFC3630] and [RFC7471].

<VN Survivability> describes all attributes related to the VN recovery level and its survivability policy enforced by the customers/applications.

```
<VN Survivability> ::= <VN Recovery Level>
 [<VN Tunnel Recovery Level>]
 [<VN Survivability Policy>]
```

Where:

<VN Recovery Level> It is a value representing the requested level of resiliency required against the VN. The following values are defined:

- . Unprotected VN
- . VN with per tunnel recovery: The recovery level is defined against the tunnels composing the VN and it is specified in the <VN Tunnel Recovery Level>.

<VN Tunnel Recovery Level> ::= <0:1>|<1+1>|<1:1>|<1:N>|<M:N>|

<On the fly restoration>

The VN Tunnel Recovery Level indicates the type of protection or restoration mechanism applied to the VN. It augments the recovery types defined in [RFC4427].

<VN Survivability Policy> ::= [<Local Reroute Allowed>]

[<Domain Preference>]

[<Push Allowed>]

[<Incremental Update>]

Where:

<Local Reroute Allowed> is a delegation policy to the Server to allow or not a local reroute fix upon a failure of the primary LSP.

<Domain Preference> is only applied on the MPI where the MDSC (client) provides a domain preference to each PNC (server).e.g. when a inter-domain link fails, then PNC can choose the alternative peering with this info.

<Push Allowed> is a policy that allows a server to trigger an updated VN topology upon failure without an explicit request from the client. Push action can be set as default unless otherwise specified.

<Incremental Update> is another policy that triggers an incremental update from the server since the last period of

update. Incremental update can be set as default unless otherwise specified.

### 2.2.3. VN End-Point

<VN End-Point> Object describes the VN's customer end-point characteristics.

```
<VN End-Point> ::= (<Access Point Identifier>
 [
 <Access Link Capability>
 [
 <Source Indicator>]]...)
```

Where:

<Access point identifier> It represents a unique identifier of the client end-point. They are used by the customer to ask for the setup of a virtual network creation. A <VN End-Point> is defined against each AP in the network and is shared between customer and provider. Both the customer and the provider will map it against his own physical resources.

<Access Link Capability> An optional object that identifies the capabilities of the access link related to the given access point. (e.g., max-bandwidth, bandwidth availability, etc.)

<Source Indicator> indicates if an End-point is source or not.

### 2.2.4. VN Objective Function

The VN Objective Function applies to each VN member (i.e., each E2E tunnel) of a VN.

The VN Objective Function can reuse objective functions defined in [RFC5541] section 4.

For a single path computation, the following objective functions are defined:

- o MCP is the Minimum Cost Path with respect to a specific metric (e.g. shortest path).



- o MLP is the Minimum Load Path, that means find a path composed by te-link least loaded.
- o MBP is the Maximum residual Bandwidth Path.

For a concurrent path computation, the following objective functions are defined:

- o MBC is to Minimize aggregate Bandwidth Consumption.
- o MLL is to Minimize the Load of the most loaded Link.
- o MCC is to Minimize the Cumulative Cost of a set of paths.

#### 2.2.5. VN Action Status

<VN Action Status> is the status indicator whether the VN has been successfully instantiated, modified, or deleted in the server network or not in response to a particular VN action.

Note that this action status object can be implicitly indicated and thus not included in any of the VN primitives discussed in Section 2.3.

#### 2.2.6. VN Associated LSP

<VN Associated LSP> describes the instantiated LSPs that is associated with the VN. <VN Associated LSP> is used between each domain PNC and the MDSC as part of VN Update once the VN is instantiated in each domain network and when CNC want to have more details about the topology instantiated as consequence of a VN Instantiate.

<VN Associated LSP> ::= <VN Identifier> (<LSP>...)

#### 2.2.7. VN Computed Path

The VN Computed Path is the list of paths obtained after the VN path computation request from higher controller. Note that the computed path is to be distinguished from the LSP. When the computed path is signaled in the network (and thus the resource is reserved for that path), it becomes an LSP.

<VN Computed Path> ::= (<Path>...)

### 2.2.8. VN Service Preference

This section provides VN Service preference. VN Service is defined in Section 2.

```
<VN Service Preference> ::= [<Location Service Preference >]
 [<Client-specific Preference >]
 [<End-Point Dynamic Selection Preference >]
```

Where

<Location Service Preference describes the End-Point Location's (e.g. Data Centers) support for certain Virtual Network Functions (VNFs) (e.g., security function, firewall capability, etc.) and is used to find the path that satisfies the VNF constraint.

<Client-specific Preference> describes any preference related to Virtual Network Service (VNS) that application/client can enforce via CNC towards lower level controllers. For example, permission the correct selection from the network of the destination related to the indicated VNF. It is e.g. the case of VM migration among data center and CNC can enforce specific policy that can permit MDSC/PNC to calculate the correct path for the connectivity supporting the data center interconnection required by application.

<End-Point Dynamic Selection Preference> describes if the End-Point (e.g. Data Center) can support load balancing, disaster recovery or VM migration and so can be part of the selection by MDSC following service Preference enforcement by CNC.

### 2.3. Mapping of VN Primitives with VN Objects

This section describes the mapping of VN Primitives with VN Objects based on Section 2.2.

```
<VN Instantiate> ::= <VN Service Characteristics>
 <VN Objective Function>
 <VN End-Point>
```

[<VN Service Preference>]

<VN Modify> ::= <VN identifier>  
                  <VN Service Characteristics>  
                  [<VN Objective Function>]  
                  <VN End-Point>  
                  [<VN Service Preference>]

<VN Delete> ::= <VN Identifier>

<VN Update> :: = <VN Identifier>  
                  <VN Associated LSP>

<VN Path Compute Request> ::= <VN Service Characteristic>  
                                  <VN Objective Function>  
                                  <VN End-Point>

<VN Path Compute Reply> ::= <VN Computed Path>

<VN Query> ::= <VN Identifier>

<VN Query Reply> ::= <VN Identifier>  
                      <VN Associated LSP>

### 3. References

#### 3.1. Normative References

[DRAFT-SER-AWARE] Dhruv Dhody, Qin Wu, Vishwas Manral, Zafar Ali, and Kenji Kumaki, "Extensions to the Path Computation Element Communication Protocol (PCEP) to compute service aware Label Switched Path (LSP).", June 2016, draft-ietf-pce-pcep-service-aware-10.

#### 3.2. Informative References

[TE-TOPO] Liu, X. et al., "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo, work in progress. Informative References

[ACTN-Req] Y. Lee, et al., "Requirements for Abstraction and Control of Transport Networks", draft-lee-teas-actn-requirements, work in progress.

[ACTN-Frame] D. Ceccarelli, et al., "Framework for Abstraction and Control of Transport Networks", draft-ceccarelli-teas-actn-framework, work in progress.

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

[RFC5541] JL. Le Roux, JP. Vasseur and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, June 2009.

[RFC7926] A. Farrel, et al., "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", RFC 7926, July 2016.

#### 4. Contributors

##### Contributors' Addresses

##### Authors' Addresses

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

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

Dhruv Dhody  
Huawei Technologies,  
Divyashree Technopark, Whitefield  
Bangalore, India  
Email: dhruv.ietf@gmail.com

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

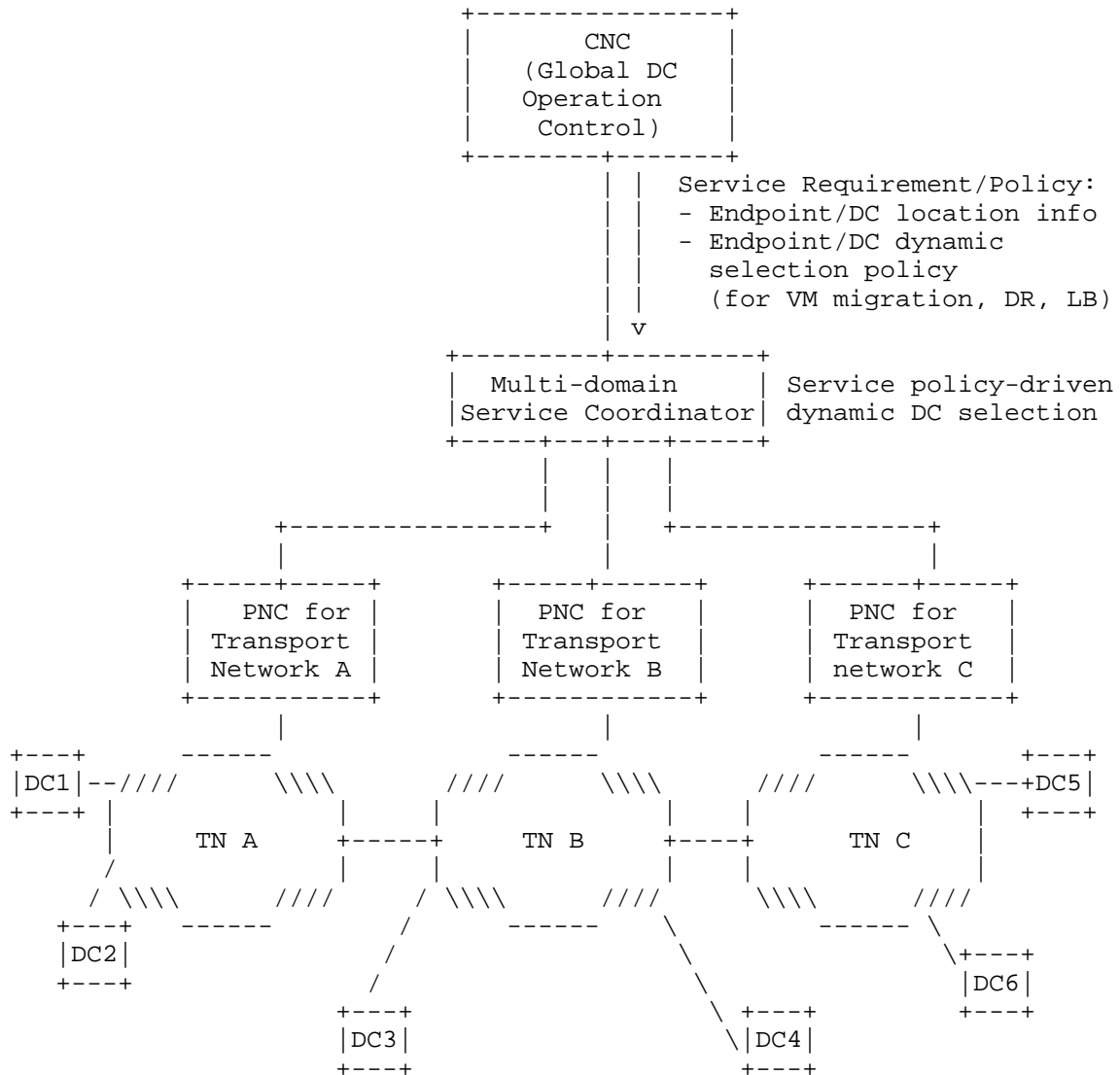
Bin Young Yun  
ETRI  
Email: byyun@etri.re.kr

Haomian Zheng  
Huawei Technologies  
Email: zhenghaomian@huawei.com

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

## Appendix A: ACTN Applications

## A.1. Coordination of Multi-destination Service Requirement/Policy



DR: Disaster Recovery

LB: Load Balancing

Figure A.1: Service Policy-driven Data Center Selection

Figure A.1 shows how VN service policies from the CNC are incorporated by the MDSC to support multi-destination applications. Multi-destination applications refer to applications in which the selection of the destination of a network path for a given source needs to be decided dynamically to support such applications.

Data Center selection problems arise for VM mobility, disaster recovery and load balancing cases. VN's service policy plays an important role for virtual network operation. Service policy can be static or dynamic. Dynamic service policy for data center selection may be placed as a result of utilization of data center resources supporting VNs. The MDSC would then incorporate this information to meet the service objective of this application.

## A.2. Application Service Policy-aware Network Operation

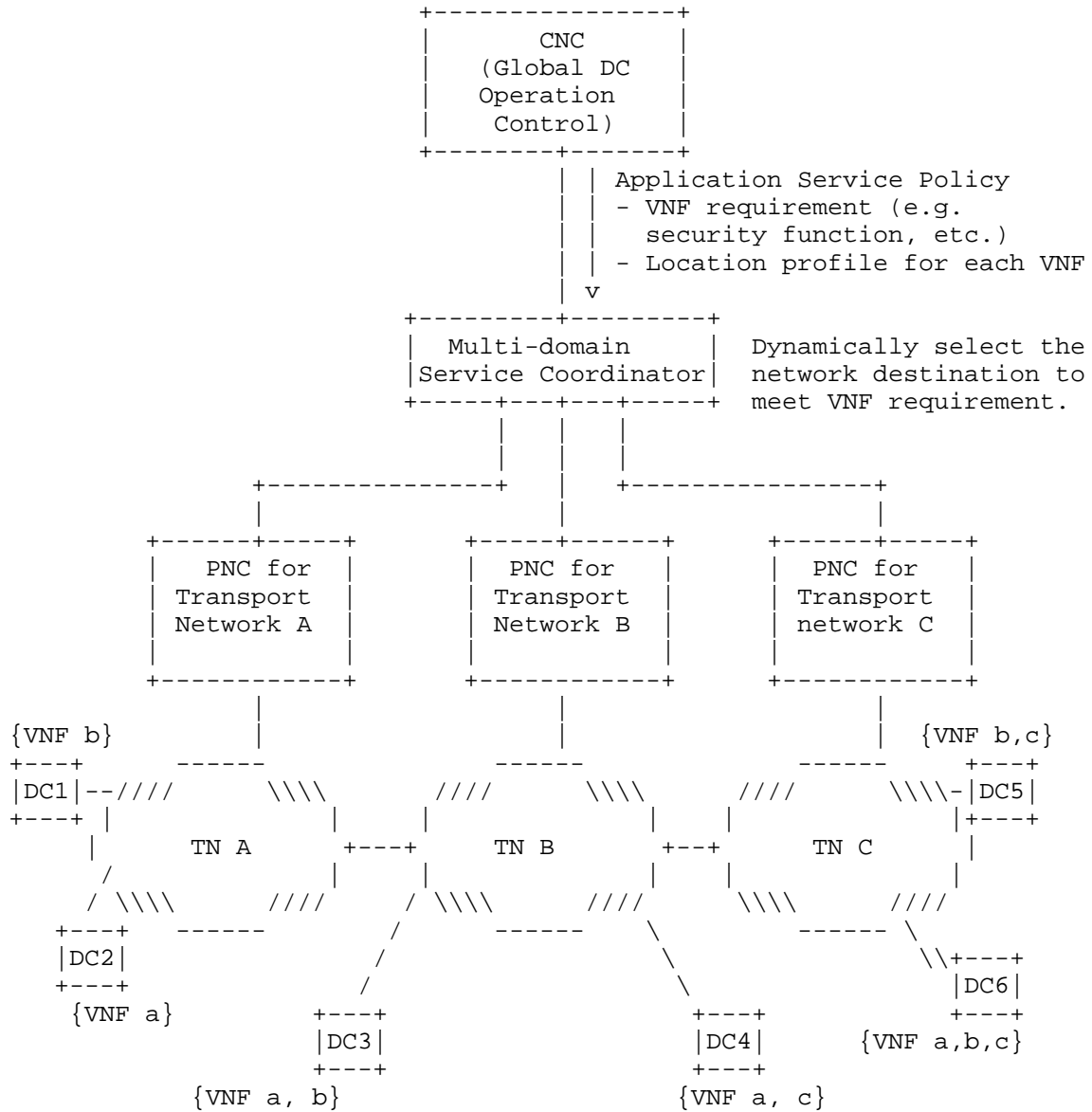


Figure A.2: Application Service Policy-aware Network Operation



This scenario is similar to the previous case in that the VN service policy for the application can be met by a set of multiple destinations that provide the required virtual network functions (VNF). Virtual network functions can be, for example, security functions required by the VN application. The VN service policy by the CNC would indicate the locations of a certain VNF that can be fulfilled. This policy information is critical in finding the optimal network path subject to this constraint. As VNFs can be dynamically moved across different DCs, this policy should be dynamically enforced from the CNC to the MDSC and the PNCs.

## A.3. Network Function Virtualization Service Enabled Connectivity

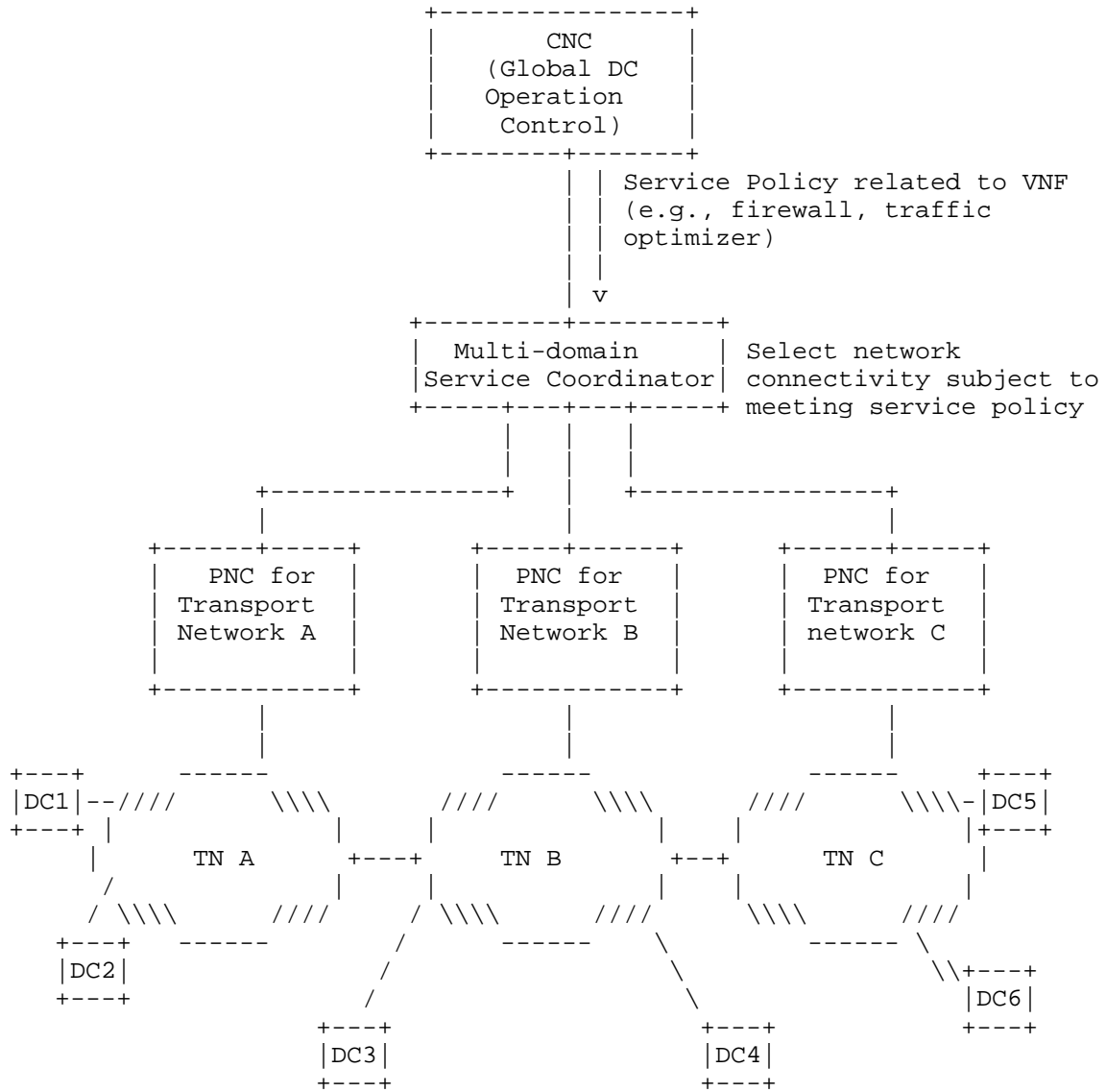


Figure A.3: Network Function Virtualization Service Enabled Connectivity

Network Function Virtualization Services are usually setup between customers' premises and service provider premises and are provided mostly by cloud providers or content delivery providers. The context may include, but not limited to a security function like firewall, a traffic optimizer, the provisioning of storage or computation capacity where the customer does not care whether the service is implemented in a given data center or another. The customer has to provide (and CNC is providing this) the type of VNF he needs and the policy associated with it (e.g. metric like estimated delay to reach where VNF is located in the DC). The policy linked to VNF is requested as part of the VN instantiation. These services may be hosted virtually by the provider or physically part of the network. This allows the service provider to hide his own resources (both network and data centers) and divert customer requests where most suitable. This is also known as "end points mobility" case and introduces new concepts of traffic and service provisioning and resiliency (e.g., Virtual Machine mobility).

#### A.4. Dynamic Service Control Policy Enforcement for Performance and Fault Management

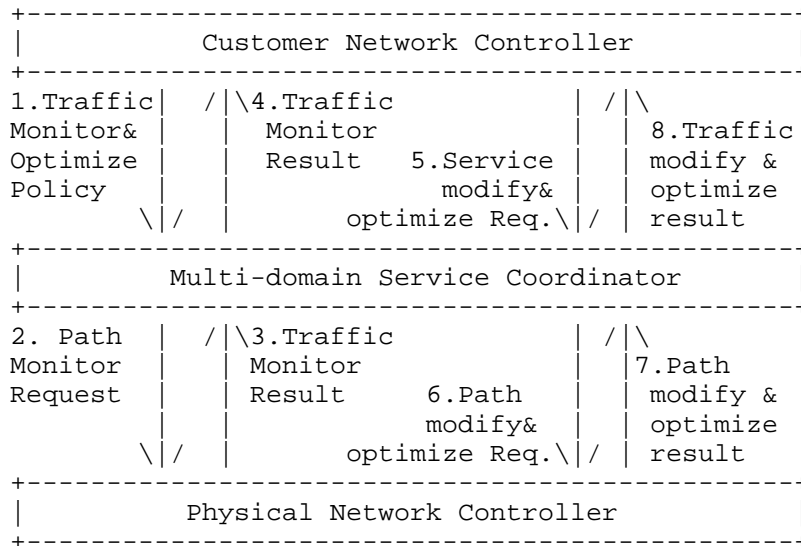


Figure A.4: Dynamic Service Control for Performance and Fault Management

Figure A.4 shows the flow of dynamic service control policy enforcement for performance and fault management initiated by customer per VN. The feedback loop and filtering mechanism tailored for VNs performed by the MDSC differentiates this ACTN scope from traditional network management paradigm. VN level dynamic OAM data model is a building block to support this capability.

#### A.5. E2E VN Survivability and Multi-Layer (Packet-Optical) Coordination for Protection/Restoration

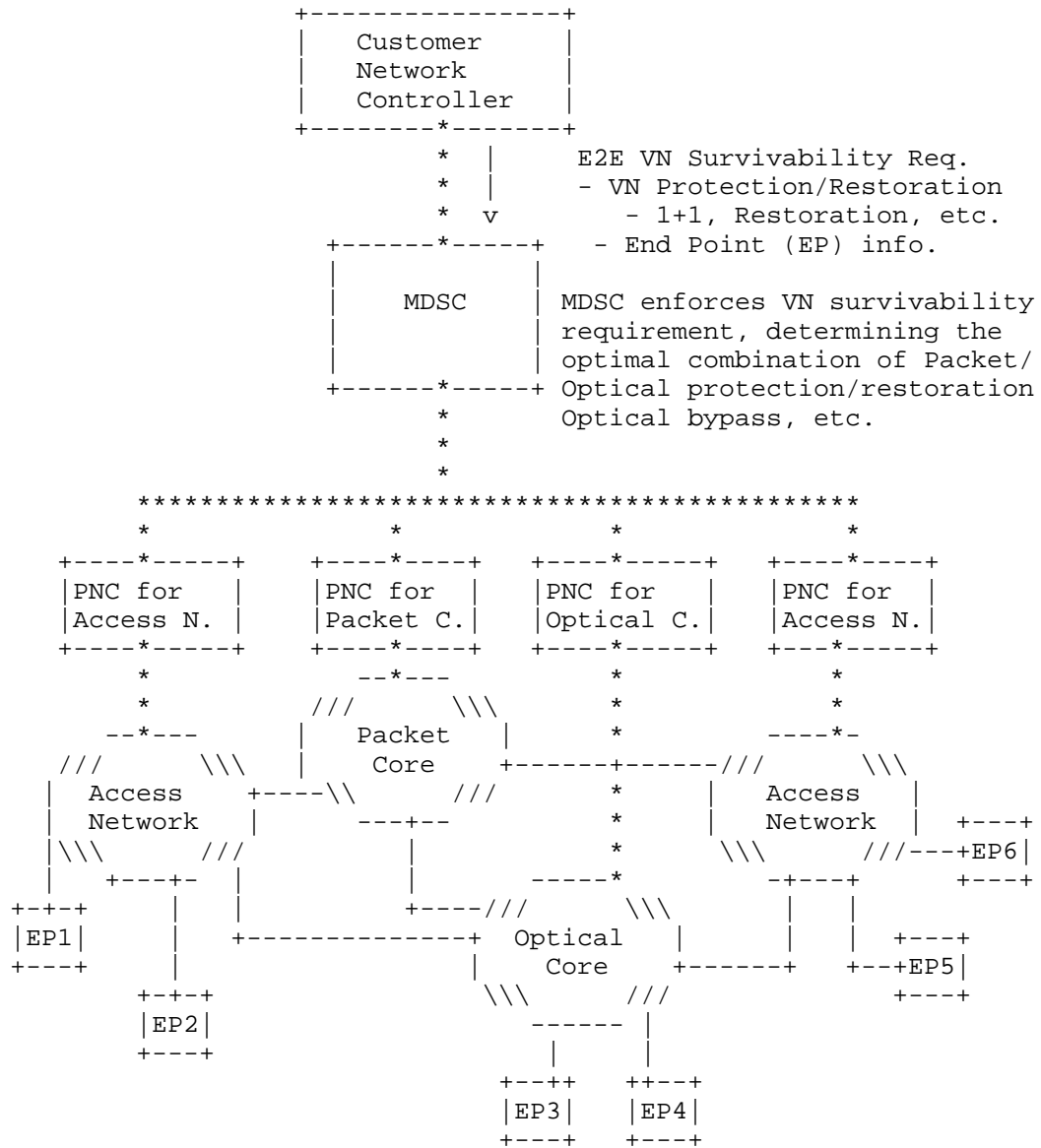


Figure A.5: E2E VN Survivability and Multi-layer Coordination for Protection and Restoration

Figure A.5 shows the need for E2E protection/restoration control coordination that involves CNC, MDSC and PNCs to meet the VN survivability requirement. VN survivability requirement and its policy need to be translated into multi-domain and multi-layer network protection and restoration scenarios across different controller types. After an E2E path is setup successfully, the MDSC has a unique role to enforce policy-based flexible VN survivability requirement by coordinating all PNC domains.

As seen in Figure A.5, multi-layer (i.e., packet/optical) coordination is a subset of this E2E protection/restoration control operation. The MDSC has a role to play in determining an optimal protection/restoration level based on the customer's VN survivability requirement. For instance, the MDSC needs to interface the PNC for packet core as well as the PNC for optical core and enforce protection/restoration policy as part of the E2E protection/restoration. Neither the PNC for packet core nor the PNC for optical core is in a position to be aware of the E2E path and its protection/restoration situation. This role of the MDSC is unique for this reason. In some cases, the MDSC will have to determine and enforce optical bypass to find a feasible reroute path upon packet core network failure which cannot be resolved the packet core network itself.

To coordinate this operation, the PNCs will need to update its domain level abstract topology upon resource changes due to a network failure or other factors. The MDSC will incorporate all these update to determine if an alternative E2E reroute path is necessary or not based on the changes reported from the PNCs. It will need to update the E2E abstract topology and the affected CN's VN topology in real-time. This refers to dynamic synchronization of topology from Physical topology to abstract topology to VN topology.

MDSC will also need to perform the path restoration signaling to the affected PNCs whenever necessary.



TEAS Working Group  
Internet-Draft  
Intended status: Informational  
Expires: August 21, 2017

H. Sitaraman, Ed.  
V. Beeram  
Juniper Networks  
I. Minei  
Google, Inc.  
S. Sivabalan  
Cisco Systems, Inc.  
February 17, 2017

Recommendations for RSVP-TE and Segment Routing LSP co-existence  
draft-sitaraman-sr-rsvp-coexistence-rec-02.txt

## Abstract

Operators are looking to introduce services over Segment Routing (SR) LSPs in networks running Resource Reservation Protocol (RSVP-TE) LSPs. In some instances, operators are also migrating existing services from RSVP-TE to SR LSPs. For example, there might be certain services that are well suited for SR and need to co-exist with RSVP-TE in the same network. In other cases, services running on RSVP-TE might be migrated to run over SR. Such introduction or migration of traffic to SR might require co-existence with RSVP-TE in the same network for an extended period of time depending on the operator's intent. The following document provides solution options for keeping the traffic engineering database (TED) consistent across the network, accounting for the different bandwidth utilization between SR and RSVP-TE.

## 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 August 21, 2017.



## 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 . . . . .                                   | 2 |
| 2. Conventions used in this document . . . . .              | 3 |
| 3. Solution options . . . . .                               | 3 |
| 3.1. Static partitioning of bandwidth . . . . .             | 3 |
| 3.2. Centralized management of available capacity . . . . . | 4 |
| 3.3. Flooding SR utilization in IGP . . . . .               | 4 |
| 3.4. Running SR over RSVP-TE . . . . .                      | 5 |
| 3.5. TED consistency by reflecting SR traffic . . . . .     | 5 |
| 4. Acknowledgements . . . . .                               | 7 |
| 5. Contributors . . . . .                                   | 7 |
| 6. IANA Considerations . . . . .                            | 8 |
| 7. Security Considerations . . . . .                        | 8 |
| 8. References . . . . .                                     | 8 |
| 8.1. Normative References . . . . .                         | 8 |
| 8.2. Informative References . . . . .                       | 8 |
| Authors' Addresses . . . . .                                | 9 |

## 1. Introduction

Introduction of SR [I-D.ietf-spring-segment-routing] in the same network domain as RSVP-TE [RFC3209] presents the problem of accounting for SR traffic and making RSVP-TE aware of the actual available bandwidth on the network links. RSVP-TE is not aware of how much bandwidth is being consumed by SR services on the network links and hence both at computation time (for a distributed computation) and at signaling time RSVP-TE LSPs will incorrectly place loads. This is true where RSVP-TE paths are distributed or centrally computed without a common entity managing both SR and RSVP-TE computation for the entire network domain.

The problem space can be generalized as a dark bandwidth problem to cases where any other service exists in the network that runs in parallel across common links and whose bandwidth is not reflected in the available and reserved values in the TED. The general problem is management of dark bandwidth pools and can be generalized to cases where any other service exists in the network that runs in parallel across common links and whose bandwidth is not reflected in the available and reserved values in the TED. In most practical instances given the static nature of the traffic demands, limiting the available reservable bandwidth available to RSVP-TE has been an acceptable solution. However, in the case of SR traffic, there is assumed to be very dynamic traffic demands and there is considerable risk associated with stranding capacity or overbooking service traffic resulting in traffic drops.

The high level requirements or assumptions to consider are:

1. Placement of SR LSPs in the same domain as RSVP-TE LSPs MUST NOT introduce inaccuracies in the TED used by distributed or centralized path computation engines.
2. Engines that compute RSVP-TE paths MAY have no knowledge of the existence of the SR paths in the same domain.
3. Engines that compute RSVP-TE paths SHOULD NOT require a software upgrade or change to their path computation logic.
4. Protocol extensions SHOULD be avoided or be minimal as in many cases this co-existence of RSVP-TE and SR MAY be needed only during a transition phase.
5. Placement of SR LSPs in the same domain as RSVP-TE LSPs that are computed in a distributed fashion MUST NOT require migration to a central controller architecture for the RSVP-TE LSPs.

## 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. Solution options

### 3.1. Static partitioning of bandwidth

In this model, the static reservable bandwidth of an interface can be statically partitioned between SR and RSVP-TE and each can operate within that bandwidth allocation and SHOULD NOT preempt each other.

While it is possible to configure RSVP-TE to only reserve up to a certain maximum link bandwidth and manage the remaining link bandwidth for other services, this is a deployment where SR and RSVP-TE are separated in the same network (ships in the night) and can lead to suboptimal link bandwidth utilization not allowing each to consume more, if required and constraining the respective deployments.

The downside of this approach is the inability to use the reservable bandwidth effectively and inability to use bandwidth left unused by the other protocol.

### 3.2. Centralized management of available capacity

In this model, a central controller performs path placement for both RSVP-TE and SR LSPs. The controller manages and updates its own view of the in-use and the available capacity. As the controller is a single common entity managing the network it can have a unified and consistent view of the available capacity at all times.

A practical drawback of this model is that it requires the introduction of a central controller managing the RSVP-TE LSPs as a prerequisite to the deployment of any SR LSPs. Therefore, this approach is not practical for networks where distributed TE with RSVP-TE LSPs is already deployed, as it requires a redesign of the network and is not backwards compatible. This does not satisfy requirement 5.

Note that it is not enough for the controller to just maintain the unified view of the available capacity, it must also perform the path computation for the RSVP-TE LSPs, as the reservations for the SR LSPs are not reflected in the TED. This does not fit with assumption 2 mentioned earlier.

### 3.3. Flooding SR utilization in IGP

Using techniques in [RFC7810], [RFC7471] and [RFC7823], the SR utilization information can be flooded in IGP-TE and the RSVP-TE path computation engine (CSPF) can be changed to consider this information. This requires changes to the RSVP-TE path computation logic and would require upgrades in deployments where distributed computation is done across the network.

This does not fit with requirements 3 and 4 mentioned earlier.

### 3.4. Running SR over RSVP-TE

SR can run over dedicated RSVP-TE LSPs that carry only SR traffic. In this model, the LSPs can be one-hop or multi-hop and can provide bandwidth reservation for the SR traffic based on functionality such as auto-bandwidth. The model of deployment would be similar in nature to running LDP over RSVP-TE. This would allow the TED to stay consistent across the network and any other RSVP-TE LSPs will also be aware of the SR traffic reservations. In this approach, non-SR traffic MUST NOT take the SR-dedicated RSVP-TE LSPs, unless required by policy.

The drawback of this solution is that it requires SR to rely on RSVP-TE for deployment. Furthermore, the accounting accuracy/frequency of this method is dependent on performance of auto-bandwidth for RSVP-TE. Note that for this method to work, the SR-dedicated RSVP-TE LSPs must be set up with the best setup and hold priorities in the network.

### 3.5. TED consistency by reflecting SR traffic

The solution relies on dynamically measuring SR traffic utilization on each TE interface and reducing the bandwidth allowed for use by RSVP-TE. It is assumed that SR traffic receives precedence in terms of the placement on the path over RSVP traffic (that is, RSVP traffic can be preempted from the path in case of insufficient resources). This is logically equivalent to SR traffic having the best preemption priority in the network. Note that this does not necessarily mean that SR traffic has higher QoS priority, in fact, SR and RSVP traffic may be in the same QoS class. The following methodology can be used at every TE node for this solution:

- o T: Traffic statistics collection time interval
- o N: Traffic averaging calculation (adjustment) interval such that  $N = k * T$ , where k is a constant integer multiplier greater or equal to 1. Its purpose is to provide a smoothing function to the statistics collection.
- o Maximum-Reservable-Bandwidth: The maximum available bandwidth for TE (this is the maximum available bandwidth on the interface, before any LSP reservations).

If Differentiated-Service (Diffserv)-aware MPLS Traffic Engineering (DS-TE) [RFC4124] is enabled, the Maximum-Reservable-Bandwidth SHOULD be interpreted as the aggregate bandwidth constraint across all Class-Types independent of the Bandwidth Constraints model.

- o RSVP-unreserved-bandwidth-at-priority-X: Maximum-Reservable-Bandwidth - sum of (existing reservations at priority X and all priorities better than X)
- o SR traffic threshold percentage: The percentage difference of traffic demand that when exceeded can result in a change to the RSVP-TE Maximum-Reservable-Bandwidth
- o IGP-TE update threshold: Specifies the frequency at which IGP-TE updates should be triggered based on TE bandwidth updates on a link
- o M: An optional multiplier that can be applied to the SR traffic average. This multiplier provides the ability to grow or shrink the bandwidth used by SR

At every interval T, each node SHOULD collect the SR traffic statistics for each of its TE interfaces. Further, at every interval N, given a configured SR traffic threshold percentage and a set of collected SR traffic statistics samples across the interval N, the SR traffic average (or any other traffic metric depending on the algorithm used) over this period is calculated.

If the difference between the new calculated SR traffic average and the current SR traffic average (that was computed in the prior adjustment) is at least SR traffic threshold percentage, then two values MUST be updated:

- o New Maximum-Reservable-Bandwidth = Current Maximum-Reservable-Bandwidth - (SR traffic average \* M)
- o New RSVP-unreserved-bandwidth-at-priority-X = New Maximum-Reservable-Bandwidth - sum of (existing reservations at priority X and all priorities better than X)

A DS-TE LSR that advertises Bandwidth Constraints TLV should update the bandwidth constraints for class-types based on operator policy. For example, when Russian Dolls Model (RDM) [RFC4127] is in use, then only BC0 may be updated. Whereas, when Maximum Allocation Model (MAM) [RFC4125] is in use, then all BCs may be updated equally such that the total value updated is equal to the newly calculated SR traffic average.

Note that the computation of the new RSVP-unreserved-bandwidth-at-priority-X MAY result in RSVP-TE LSPs being hard or soft preempted. Such preemption will be based on relative priority (e.g. low to high) between RSVP-TE LSPs. It is RECOMMENDED that the IGP-TE update threshold SHOULD be lower in order to flood unreserved bandwidth

updates often. From an operational point of view, an implementation SHOULD be able to expose both the configured and the actual values of the Maximum-Reservable-Bandwidth.

If LSP preemption is not acceptable, then the RSVP-TE Maximum-Reservable-Bandwidth cannot be reduced below what is currently reserved by RSVP-TE on that interface. This may result in bandwidth not being available for SR traffic. Thus, it is required that any external controller managing SR LSPs SHOULD be able to detect this situation (for example by subscribing to TED updates [RFC7752]) and SHOULD take action to reroute existing SR paths.

Generically, SR traffic (or any non-RSVP-TE traffic) should have its own priority allocated from the available priorities. This would allow SR to preempt other traffic according to the preemption priority order.

In this solution, the logic to retrieve the statistics, calculating averages and taking action to change the Maximum-Reservable-Bandwidth is an implementation choice, and all changes are local in nature. However, note that this is a new network trigger for RSVP-TE preemption and thus is a consideration for the operator.

The above solution offers the advantage of not introducing new network-wide mechanisms especially during scenarios of migrating to SR in an existing RSVP-TE network and reusing existing protocol mechanisms.

#### 4. Acknowledgements

The authors would like to thank Steve Ulrich for his detailed review and comments.

#### 5. Contributors

The following individuals contributed to this document:

Chandra Ramachandran  
Juniper Networks  
Email: csekar@juniper.net

Raveendra Torvi  
Juniper Networks  
Email: rtorvi@juniper.net

Sudharsana Venkataraman  
Juniper Networks  
Email: sudharsana@juniper.net

## 6. IANA Considerations

This draft does not have any request for IANA.

## 7. Security Considerations

No new security issues are introduced in this document beyond is already part of RSVP-TE and Segment routing architectures.

## 8. References

### 8.1. Normative References

- [I-D.ietf-spring-segment-routing]  
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S.,  
and R. Shakir, "Segment Routing Architecture", draft-ietf-  
spring-segment-routing-11 (work in progress), February  
2017.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V.,  
and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP  
Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001,  
<<http://www.rfc-editor.org/info/rfc3209>>.

### 8.2. Informative References

- [RFC4124] Le Faucheur, F., Ed., "Protocol Extensions for Support of  
Diffserv-aware MPLS Traffic Engineering", RFC 4124,  
DOI 10.17487/RFC4124, June 2005,  
<<http://www.rfc-editor.org/info/rfc4124>>.
- [RFC4125] Le Faucheur, F. and W. Lai, "Maximum Allocation Bandwidth  
Constraints Model for Diffserv-aware MPLS Traffic  
Engineering", RFC 4125, DOI 10.17487/RFC4125, June 2005,  
<<http://www.rfc-editor.org/info/rfc4125>>.
- [RFC4127] Le Faucheur, F., Ed., "Russian Dolls Bandwidth Constraints  
Model for Diffserv-aware MPLS Traffic Engineering",  
RFC 4127, DOI 10.17487/RFC4127, June 2005,  
<<http://www.rfc-editor.org/info/rfc4127>>.

- [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, <<http://www.rfc-editor.org/info/rfc7471>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<http://www.rfc-editor.org/info/rfc7752>>.
- [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, <<http://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, <<http://www.rfc-editor.org/info/rfc7823>>.

#### Authors' Addresses

Harish Sitaraman (editor)  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA 94089  
US

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

Vishnu Pavan Beeram  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
US

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



Ina Minei  
Google, Inc.  
1600 Amphitheatre Parkway  
Mountain View, CA 94043  
US

Email: inaminei@google.com

Siva Sivabalan  
Cisco Systems, Inc.  
2000 Innovation Drive  
Kanata, Ontario K2K 3E8  
Canada

Email: msiva@cisco.com

TEAS Working Group  
Internet Draft

A.Wang  
China Telecom

Intended status: Standard Track  
Expires: December 30, 2016

June 30, 2016

PCE in Native IP Network  
draft-wang-teas-pce-native-ip-00.txt

Status of this Memo

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, and it may not be published except as an Internet-Draft.

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English.

it for publication as an RFC or to translate it into languages other than English.

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 1, 2009.

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

## Abstract

This document defines the PCE use case and solution that can be deployed within the native IP network, using Multi-BGP session strategy and PCE-based central control to assure the end2end traffic performance, and proposes the corresponding extension to PCEP protocol to transfer the key parameters between PCE and the underlying network device (PCC).

## Table of Contents

|                                                              |   |
|--------------------------------------------------------------|---|
| 1. Introduction .....                                        | 2 |
| 2. Conventions used in this document .....                   | 3 |
| 3. Dual-BGP solution for simple topology.....                | 3 |
| 4. Dual-BGP in large Scale Topology .....                    | 5 |
| 5. Multi-BGP for Extended Traffic Differentiation .....      | 5 |
| 6. PCE based solution for Multi-BGP strategy deployment..... | 6 |
| 7. PCEP extension for key parameter transformation. ....     | 8 |
| 8. Security Considerations .....                             | 8 |
| 9. IANA Considerations .....                                 | 8 |
| 10. Conclusions .....                                        | 8 |
| 11. References .....                                         | 8 |
| 11.1. Normative References .....                             | 8 |
| 11.2. Informative References.....                            | 9 |
| 12. Acknowledgments .....                                    | 9 |

## 1. Introduction

Currently, PCE based traffic assurance requires the underlying network devices support MPLS and the network must deploy multiple LSPs to assure the end-to-end traffic performance. LDP/RSVP-TE or Segment Routing should be enabled within the network to establish various MPLS paths. Such solution will certainly work but the main drawback of it is that all the LSP paths are divided logically, that is to say, all the LSP paths that go through one physical link will share and compete the

Internet-Draft                      PCE in Native IP Network                      June 30, 2016  
same resource and MPLS technology has no better solution to meet the requirements for determined QoS effect.

On the other hand, there are some legacy networks that does not deploy the MPLS control and forward plane technology, but also need to assure the QoS of application traffic. Deploy some dedicated links statically to meet such requirements is one option but it is not feasible in the service provider network, because the volume and path of application traffic will be vary from time to time.

In summary, there are scenarios that the current PCE-based MPLS solution can't be deployed within the network, because the following user requirements:

- 1)              End to End traffic assurance.
- 2)              Determined Qos Effect.
- 3)              No complex MPLS signaling procedure, support Native IP environment.
- 4)              Flexible deployment
- 5)              Central control.

This document defines the PCE use case and solution that can be deployed within the native IP network, using PCE-based central control and Multi BGP sessions strategy to assure the end2end traffic performance, meet the above requirements in dynamical and central control mode, proposes the corresponding extension to PCEP protocol to transfer the key parameters between PCE and the underlying network device(PCC).

## 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 solution for simple topology.

This section introduces first the dual-BGP solution for simple topology that 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 are normal traffic, traffic between IP12 and IP22 are priority traffic that should be treated differently. There is only Native IP protocol being deployed between R1 and R2. The traffic between each address pair will be changed timely and the corresponding source/destination addresses of the traffic may also be changed dynamically.

The key idea of the Dual-BGP solution for this simple topology is the

- 1) Build two BGP sessions between R1 and R2, via the different loopback address lo0,lo1 on these routers.
- 2) 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).
- 3) Set the static 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 IP12, and the traffic between IP21 and IP22 will go through different physical links between R1 and R2, each type of traffic occupied the different dedicated physical links and will not influence with each other.

If there is more traffic between IP12 and IP13 need to be assured, one can reassign more physical links on R1 and R2 to reach the loopback address lo1(also the next hop for BGP Peer pair2), the prefixed that advertised by two BGP peer need not be changed.

If, for example, there are traffic from another address pair need 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.

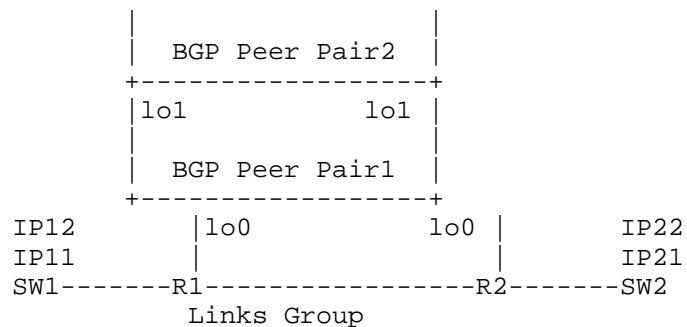


Fig.1 Design Philosophy for Dual-BGP Solution

#### 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 neighbor by neighbor 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, that is to say, select one router which performs the role of RR (for example R3 in Fig.2 - Dual-BGP Solution using Route Reflector for large scale network), every other router will establish two BGP 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 static routes on these router respectively, point 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

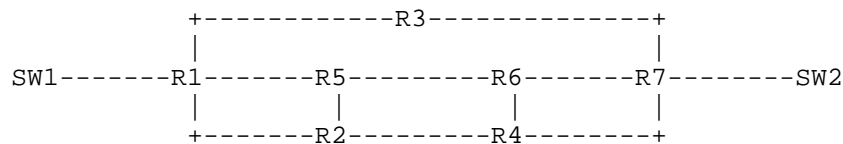


Fig.2 Dual-BGP solution using route reflector for large scale network

#### 5. Multi-BGP for Extended Traffic Differentiation

Discussed in the document so far, is the requirement for traffic differentiation to classify traffic into two classes: Assured traffic or best effort (normal) traffic. Dual-BGP solution (simple topology or large scale topology) can meet above requirements. In general situations, several additional traffic differentiation criteria exist, including:

- ? Traffic requires low latency links and not sensitive to packet loss
- ? Traffic requires low packet loss but can endure higher latency
- ? Traffic requires lowest jitter path

? Traffic requires high bandwidth links

These varying traffic requirements may be summarized in the following table:

| Flow No. | Latency | Packet Loss | Jitter      |
|----------|---------|-------------|-------------|
| 1        | Low     | Normal      | Don't care  |
| 2        | Normal  | Low         | Dont'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 composition requirements in large scale network via the traditional distributed routing protocol, but these requirements can be solved using the PCE-based 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. PCE based solution for Multi-BGP strategy deployment.

With the advent of SDN concepts within IP network, it is possible to deploy the PCE related technology into the underlying native IP network, 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:

- 1) PCE gets underlying topology information via the BGP-LS protocol via one router, such as the route reflector R3 in Fig.3
- 2) It collects also the link utilization information via the SNMP protocol.

- 3) Upon the application's requirement, for example, the bi-direction traffic assurance between SW1/SW2, the PCE will calculate the appropriate link path, which can be assigned to such traffic in dedicated mode, other normal traffic will not pass through such physical links.
- 4) PCE will then send the key parameters to R1 and R7 respectively, to let R1 and R7 build another i/eBGP neighbor with R3, advertise the prefixes that owned by SW1/SW2.
- 5) If the calculated dedicated path is via some physical links that belong to R1-R2-R4-R7, then PCE need also build the connection with these on-path routers, send some key parameters to them to build the path to the BGP next-hop via the address of physical links between R1/R2,R2/R4,R4/R7.
- 6) If the assured traffic prefixes are changed and the total volume of assured traffic is not exceed the physical capacity of the previous end-to-end path, then the PCE need only change the related information on R1 and R7.
- 7) If the volume of the assured traffic exceeds the capacity of previous calculated path, then PCE must recalculate the appropriate path to accommodate the exceeding traffic in some new end-to-end physical link. It then need to send some relevant key parameters to the on-path routers to build such path hop by hop.

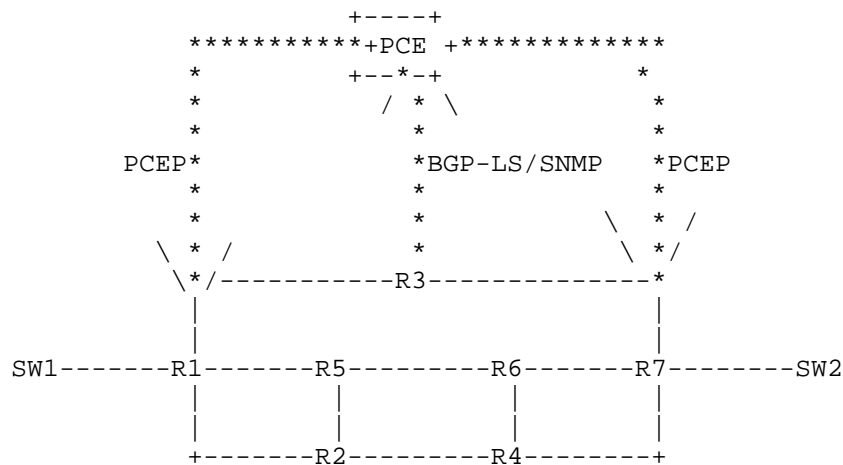


Fig.3 PCE based solution for Multi-BGP deployment



## 7. PCEP extension for key parameter transformation.

In order to pass the key parameters to the underlying routers and keep the overall implementation as simple as possible, it is appropriate to extend the PCEP protocol to transfer the key parameters.

Based on the design philosophy of afore mentioned Multi-BGP deployment scenario, the key parameters should include the following information:

- 1) BGP peer address and assured prefixes that will be advertised via this BGP session
- 2) Static route information/Destination(BGP next hop) and Next Physical Link Address.

Once the router receive such information, it should establish the BGP session with the peer appointed in the PCEP message, advertises the prefixes that contained in the corresponding PCEP message, and build the end to end dedicated path hop by hop.

The detail format and the processing procedure of the above two extensions will be provided in another draft.

## 8. Security Considerations

TBD

## 9. IANA Considerations

TBD

## 10. Conclusions

TBD

## 11. References

### 11.1. Normative References

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006, <<http://www.rfc-editor.org/info/rfc4655>>.

[RFC5440] Vasseur, JP., Ed., and JL. Le Roux, Ed., "Path

(PCEP)", RFC 5440, March 2009,

<<http://www.rfc-editor.org/info/rfc5440>>.

## 11.2. Informative References

TBD

## 12. Acknowledgments

TBD

## Authors' Addresses

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

Email: [wangaj@ctbri.com.cn](mailto:wangaj@ctbri.com.cn)



TEAS Working Group  
Internet Draft

A.Wang  
China Telecom  
Quintin Zhao  
Boris Khasanov  
HuaiMo Chen  
Huawei Technologies  
Penghui Mi  
Tencent Company

Intended status: Experimental Track  
Expires: July 24, 2018

January 25, 2018

PCE in Native IP Network  
draft-wang-teas-pce-native-ip-07.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). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on July 24, 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.

#### Abstract

This document defines the framework for CCDR traffic engineering within Native IP network, using Dual/Multi-BGP session strategy and PCE-based central control architecture.

<A.Wang>

Expires July 24, 2018

[Page 1]

Internet-Draft                      PCE in Native IP Network                      January 25, 2017  
The proposed central mode control framework conforms to the concept  
that defined in RFC " An Architecture for Use of PCE and the PCE  
Communication Protocol (PCEP) in a Network with Central Control".

The scenario and simulation results of CCDD traffic engineering is  
described in draft "CCDD Scenario, Simulation and Suggestion".

## Table of Contents

|                                                                |    |
|----------------------------------------------------------------|----|
| 1. Introduction .....                                          | 2  |
| 2. Dual-BGP framework for simple topology. ....                | 3  |
| 3. Dual-BGP in large Scale Topology .....                      | 4  |
| 4. Multi-BGP for Extended Traffic Differentiation .....        | 5  |
| 5. CCDD based framework for Multi-BGP strategy deployment..... | 6  |
| 6. PCEP extension for key parameters delivery. ....            | 7  |
| 7. CCDD Deployment Consideration .....                         | 7  |
| 8. Security Considerations.....                                | 8  |
| 9. IANA Considerations .....                                   | 8  |
| 10. Conclusions .....                                          | 8  |
| 11. References .....                                           | 9  |
| 11.1. Normative References.....                                | 9  |
| 11.2. Informative References.....                              | 9  |
| 12. Acknowledgments .....                                      | 10 |

## 1. Introduction

Draft [I-D.draft-wang-teas-ccdd] describes the scenario and simulation  
results for the CCDD traffic engineering. In summary, the requirements for  
CCDD traffic engineering in Native IP network are the following:

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

This document defines the framework for CCDD traffic engineering  
within Native IP network, using Dual/Multi-BGP session strategy and  
CCDD 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 [draft-wang-pcep-extension-native-IP]

## 2. 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:

- 1) Build two BGP sessions between R1 and R2, via the different loopback address lo0, lo1 on these routers.
- 2) 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).
- 3) 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 lo1(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.

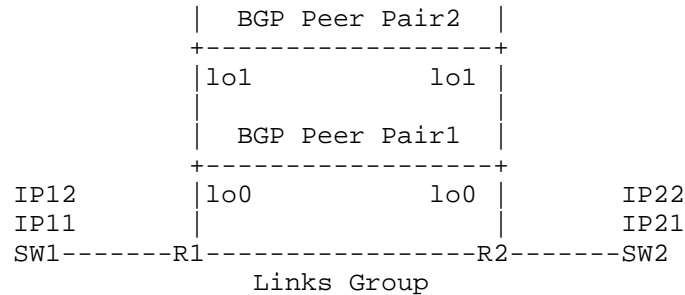
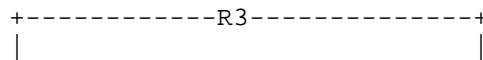


Fig.1 Design Philosophy for Dual-BGP Framework

### 3. 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



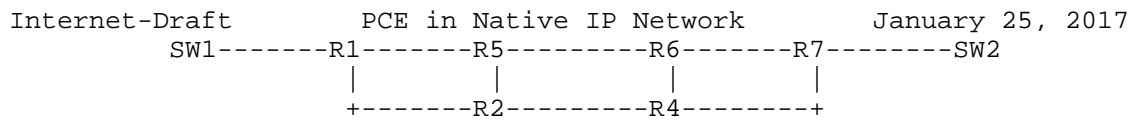


Fig.2 Dual-BGP Framework for large scale network

#### 4. 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         | Dont'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



## 5. 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:

- 1) PCE gets topology and link utilization information from the underlying network, calculate the appropriate link path upon application's requirements.
- 2) 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.
- 3) 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.
- 4) 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).
- 5) 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.

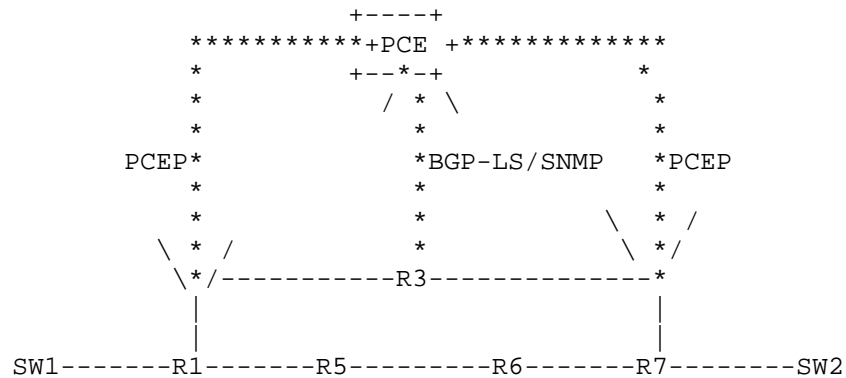




Fig.3 PCE based framework for Multi-BGP deployment

## 6. PCEP extension for key parameters delivery.

The PCEP protocol needs to be extended to transfer the following key parameters:

- 1) BGP peer address and advertised prefixes.
- 2) Explicit route information to BGP next hop of advertised prefixes.

Once the router receives such information, it should establish the BGP session with the peer appointed in the PCEP message, advertise the prefixes that contained in the corresponding PCEP message, and build the end to end dedicated path hop by hop. Details of communications between PCEP and BGP subsystems in router's control plane are out of scope of this draft and will be described in separate draft.[draft-wang-pce-extension for native IP]

The reason why we selected PCEP as the southbound protocol instead of OpenFlow, is that PCEP is suitable for the changes in control plane of the network devices, there OpenFlow dramatically changes the forwarding plane. We also think that the level of centralization that requires by OpenFlow is hardly achievable in many today's SP networks so hybrid BGP+PCEP approach looks much more interesting.

## 7. CCDR Deployment Consideration

CCDR framework requires the parallel work of 2 subsystems in router's control plane: PCE (PCEP) and BGP as well as coordination between them, so it might require additional planning work before 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

Internet-Draft                      PCE in Native IP Network                      January 25, 2017  
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 [draft-wang-pce-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.

## 8. Security Considerations

TBD

## 9. IANA Considerations

TBD

## 10. Conclusions

TBD

#### 11.1. Normative References

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006, <<http://www.rfc-editor.org/info/rfc4655>>.

[RFC5440] Vasseur, JP., Ed., and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, March 2009, <<http://www.rfc-editor.org/info/rfc5440>>.

[RFC8283] A.Farrel, Q.Zhao et al., "An Architecture for Use of PCE and the PCE Communication Protocol (PCEP) in a Network with Central Control", [RFC8283], December 2017

#### 11.2. Informative References

[I-D.draft-wang-teas-ccdr]

A.Wang, X.Huang et al. "CCDR Scenario, Simulation and Suggestion" <https://datatracker.ietf.org/doc/draft-wang-teas-ccdr/>

[I-D. draft-ietf-teas-pcecc-use-cases]

Quintin Zhao, Robin Li, Boris Khasanov et al. "The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs" <https://tools.ietf.org/html/draft-ietf-teas-pcecc-use-cases-00>  
March, 2017

[draft-wang-pcep-extension for native IP]

## 12. Acknowledgments

The authors would like to thank George Swallow, Xia Chen, Jeff Tantsura, Scharf Michael, Daniele Ceccarelli and Dhruv Dhody for their valuable comments and suggestions.

The authors would also like to thank Lou Berger, Adrian Farrel, Vishnu Pavan Beeram, Deborah Brungard and King Daniel for their suggestions to put forward this draft.

## Authors' Addresses

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

Email: wangaj.bri@chinatelecom.cn

Internet-Draft                      PCE in Native IP Network

January 25, 2017

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  
Tencent  
Tencent Building, Kejizhongyi Avenue,  
Hi-techPark, Nanshan District, Shenzhen 518057, P.R.China

Email kevinmi@tencent.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: Informational

Expires: December 31, 2017

Young Lee  
Haomian Zheng  
Huawei

Daniel Ceccarrelli  
Ericsson

Bin Yeong Yoon  
ETRI

Oscar Gonzalez de Dios  
Telefonica

Jong Yoon Shin  
SKT

Sergio Belotti  
Nokia

June 30, 2017

Applicability of YANG models for Abstraction and Control of Traffic  
Engineered Networks

draft-zhang-teas-actn-yang-05

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 30, 2017.

#### 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.

#### Abstract

Abstraction and Control of TE Networks (ACTN) refers to the set of virtual network operations needed to orchestrate, control and manage large-scale multi-domain TE networks, so as to facilitate network programmability, automation, efficient resource sharing, and end-to-end virtual service aware connectivity and network function virtualization services.

This document explains how the different types of YANG models defined in the Operations and Management Area and in the Routing Area are applicable to the ACTN framework. This document also shows how the ACTN architecture can be satisfied using classes of data model that have already been defined, and discusses the applicability of specific data models that are under development. It also highlights where new data models may need to be developed.

#### Table of Contents

|                                                                    |   |
|--------------------------------------------------------------------|---|
| 1. Introduction.....                                               | 3 |
| 2. Abstraction and Control of TE Networks (ACTN) Architecture..... | 3 |
| 3. Service Models.....                                             | 5 |
| 4. Service Model Mapping to ACTN.....                              | 6 |

|                                                                 |    |
|-----------------------------------------------------------------|----|
| 4.1. Customer Service Models in the ACTN Architecture (CMI).... | 7  |
| 4.2. Service Delivery Models in ACTN Architecture.....          | 8  |
| 4.3. Network Configuration Models in ACTN Architecture (MPI)... | 8  |
| 4.4. Device Models in ACTN Architecture (SBI).....              | 9  |
| 5. Examples of Using Different Types of YANG Models.....        | 10 |
| 5.1. Simple Connectivity Examples.....                          | 10 |
| 5.2. VN service example.....                                    | 10 |
| 5.3. Data Center-Interconnection Example.....                   | 11 |
| 5.3.1. CMI (CNC-MDSC Interface).....                            | 13 |
| 5.3.2. MPI (MDSC-PNC Interface).....                            | 13 |
| 5.3.3. PDI (PNC-Device interface).....                          | 13 |
| 6. Security.....                                                | 14 |
| 7. Acknowledgements.....                                        | 14 |
| 8. References.....                                              | 14 |
| 8.1. Informative References.....                                | 14 |
| 9. Contributors.....                                            | 16 |
| Authors' Addresses.....                                         | 17 |

## 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 transport network) to provide connectivity and virtual network services for customers of the TE network. The services provided can be tuned to meet the requirements (such as traffic patterns, quality, and reliability) of the applications hosted by the customers. More details about ACTN can be found in Section 2.

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 modelling 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].

This document shows how the ACTN architecture can be satisfied using classes of data model that have already been defined, and discusses the applicability of specific data models that are under development. It also highlights where new data models may need to be developed.

## 2. Abstraction and Control of TE Networks (ACTN) Architecture

[ACTN-Requirements] describes the high-level ACTN requirements.

[ACTN-Frame] describes the architecture model for ACTN including the

entities (Customer Network Controller (CNC), Multi-domain Service Coordinator (MDSC), and Physical Network Controller (PNC)) and their interfaces.

Figure 1 depicts a high-level control and interface architecture for ACTN and is a reproduction of Figure 3 from [ACTN-Frame]. A number of key ACTN interfaces exist for deployment and operation of ACTN-based networks. These are highlighted in Figure 1 (ACTN Interfaces) below:

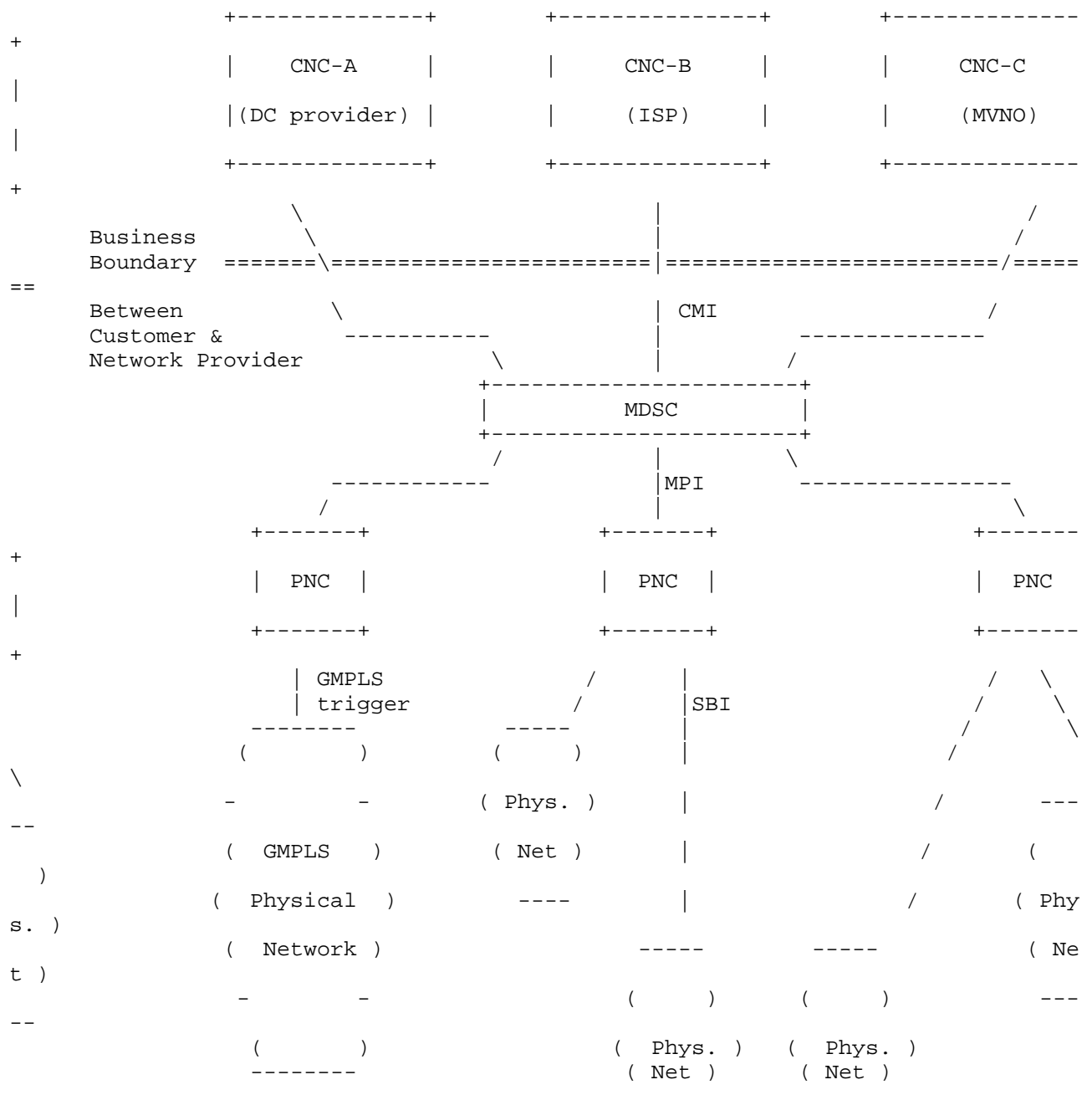


Figure 1 : ACTN Interfaces

The interfaces and functions are described below (without modifying the definitions) in [ACTN-Frame]:



- . The CNC-MDSC Interface (CMI) is an interface between a Customer Network Controller and a Multi Domain Service Controller. The interface will communicate the service request or application demand. A request will include specific service properties, for example, services type, bandwidth and constraint information. These constraints SHOULD be measurable by MDSC and therefore visible to CNC via CMI. The CNC can also request the creation of the virtual network based on underlying physical resources to provide network services for the applications. The CNC can provide the end-point information/characteristics, traffic matrix specifying specific customer constraints. The MDSC may also report potential network topology availability if queried for current capability from the Customer Network Controller.
- . The MDSC-PNC Interface (MPI) is an interface between a Multi Domain Service Coordinator and a Physical Network Controller. It allows the MDSC to communicate requests to create/delete connectivity or to modify bandwidth reservations in the physical network. In multi-domain environments, each PNC is responsible for a separate domain. The MDSC needs to establish multiple MPIs, one for each PNC and perform coordination between them to provide cross-domain connectivity.
- . The South-Bound Interface (SBI) is the provisioning interface for creating forwarding state in the physical network, requested via the Physical Network Controller. The SBI is not in the scope of ACTN, however, it is included in this document so that it can be compared to models in [Service-Yang].

### 3. Service Models

[Service-YANG] introduces a reference architecture to explain the nature and usage of service YANG models in the context of service orchestration. Figure 2 below depicts this relationship and is a reproduction of Figure 2 from [Service-YANG]. Four models depicted in Figure 2 are defined as follows:

- . Customer Service Model: A customer service model is used to describe a service as offer or delivered to a customer by a network operator.
- . Service Delivery Model: A service delivery model is used by a network operator to define and configure how a service is provided by the network.
- . Network Configuration Model: A network configuration model is used by a network orchestrator to provide network-level configuration model to a controller.

- . Device Configuration Model: A device configuration model is used by a controller to configure physical network elements.

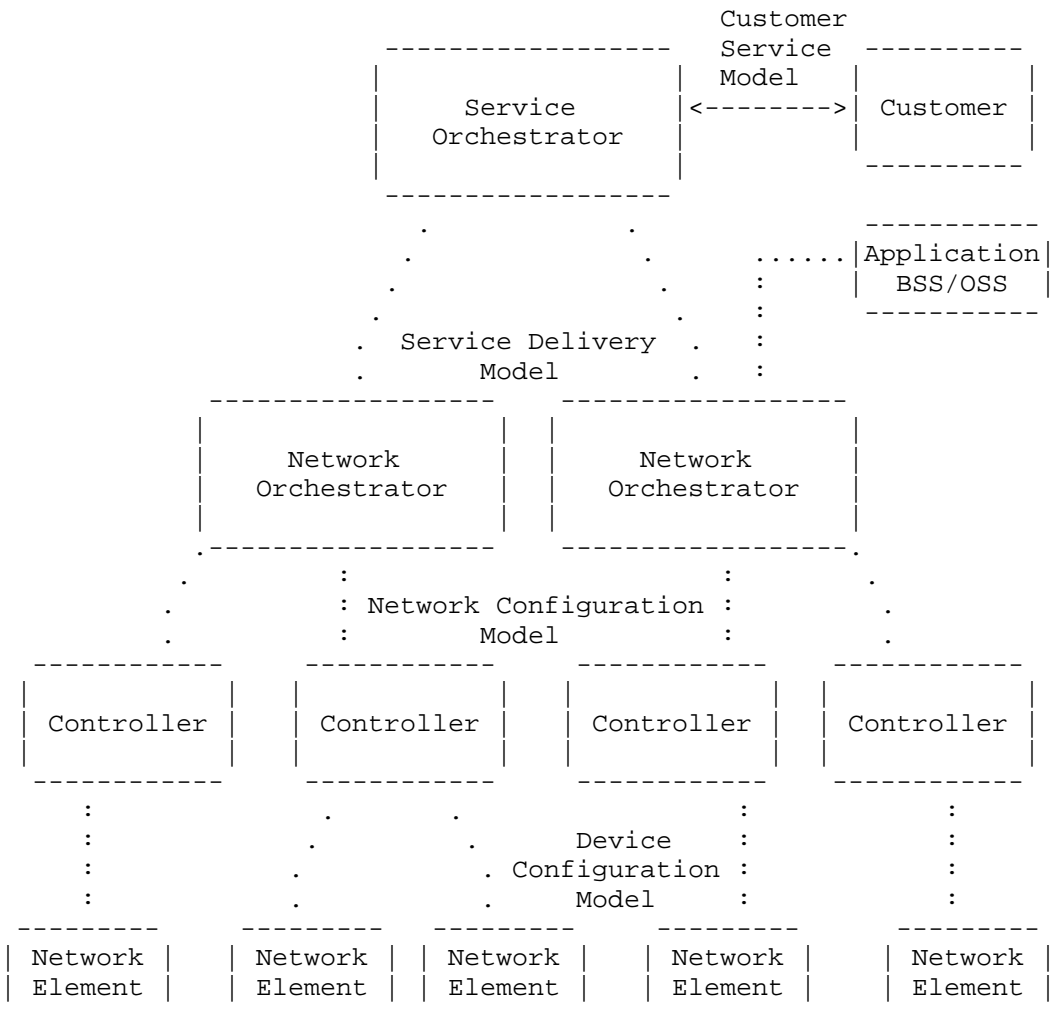


Figure 2: An SDN Architecture with a Service Orchestrator

#### 4. Service Model Mapping to ACTN

YANG models coupled with the RESTCONF/NETCONF protocol [Netconf][Restconf] provides solutions for the ACTN framework. This section explains which types of YANG models apply to each of the ACTN interfaces.

Refer to Figure 5 of [ACTN-Frame] for details of the mapping between ACTN functions and service models. In summary, the following mappings are held between Service Yang Models and the ACTN interfaces.

- o Customer Service Model <-> CMI
- o Network Configuration Model <-> MPI
- o Device Configuration Model <-> SBI

#### 4.1. Customer Service Models in the ACTN Architecture (CMI)

Customer Service Models, which are used between a customer and a service orchestrator as in [Service-YANG], should be used between the CNC and MDSC (e.g., CMI) serving as providing a simple intent-like model/interface.

Among the key functions of Customer Service Models on the CMI is the service request. A request will include specific service properties, including: service type and its characteristics, bandwidth, constraint information, and end-point characteristics.

The following table provides a list of functions needed to build the CMI. They are mapped with Customer Service Models.

| Function                            | Yang Model                |
|-------------------------------------|---------------------------|
| Transport Service Request           | [Transport-Service-Model] |
| VN Service Request & Instantiation  | [ACTN-VN-YANG]            |
| VN Path Computation Request         | [ACTN-VN-YANG]*           |
| VN Performance Monitoring Telemetry | [ACTN-PM-Telemetry]**     |
| Topology Abstraction                | [TE-topology]             |

\*VN Path computation request in the CMI context means network path computation request based on customer service connectivity request constraints prior to the instantiation of a VN creation.

\*\*ietf-actn-te-kpi-telemetry model describes performance telemetry for ACTN VN model. 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.

#### 4.2. Service Delivery Models in ACTN Architecture

The Service Delivery Models where the service orchestration and the network orchestration could be implemented as separate components as seen in [Service-YANG]. This is also known as Network Service Models. On the other hand, from an ACTN architecture point of view, the service delivery model between the service orchestrator and the network orchestrator is an internal interface between sub-components of the MDSC in a single MDSC model.

In the MDSC hierarchical model where there are multiple MDSCs, the interface between the top MDSC and the bottom MDSC can be mapped to service delivery models.

#### 4.3. Network Configuration Models in ACTN Architecture (MPI)

The Network Configuration Models is used between the network orchestrator and the controller in [Service-YANG]. In ACTN, this model is used primarily between a MDSC and a PNC. The Network Configuration Model can be also used for the foundation of more advanced models, like hierarchical MDSCs (see Section 4.5)

The Network Configuration Model captures the parameters which are network wide information.

The following table provides a list of functions needed to build the MPI. They are mapped with Network Configuration Yang Models. Note that various Yang models are work in progress.

| Function                        | Yang Model              |
|---------------------------------|-------------------------|
| Configuration Scheduling        | [Schedule]              |
| Path computation                | [PATH_COMPUTATION-API]* |
| Path Provisioning               | [TE-Tunnel]**           |
| Topology Abstraction            | [TE-topology]           |
| Tunnel PM Telemetry             | [ACTN-PM-Telemetry]***  |
| Service Provisioning            | TBD****                 |
| OTN Topology Abstraction        | [OTN-YANG]              |
| WSON Topology Abstraction       | [WSON-YANG]             |
| Flexi-grid Topology Abstraction | [Flexi-YANG]            |
| ODU Tunnel Model                | [ODU-Tunnel]            |



|                         |                    |
|-------------------------|--------------------|
| WSON TE Tunnel Model    | [WSON-Tunnel]      |
| Flexi-grid Tunnel Model | [Flexigrid-Tunnel] |

\* Related draft is presenting use cases for path computation API, and Yang related model is foreseen to be added.

\*\* Note that path provisioning function is provided by ietf-te module in [TE-Tunnel].

\*\* ietf-actn-te-kpi-telemetry model describes performance telemetry for TE-tunnel model. 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.

\*\*\*\* This function needs to be investigated further. This can be a part of [TE-Tunnel] which is to be determined. Service provisioning is an optional function that builds on top the path provisioning one.

Path provisioning and Topology abstraction functions are mandatory in any case, while Path Computation may be mandatory or optional depending on the type of topology abstraction used. Details of this topic are discussed in [ACTN-Abstraction].

Telemetry may also be an optional function.

#### 4.4. Device Models in ACTN Architecture (SBI)

For the device YANG models are used for per-device configuration purpose, they can be used between the PNC and the physical network/devices. Note that SBI is not in the scope of ACTN. This section is provided to give some examples of YANG-based Device Models. An example of Device Models is ietf-te-device yang module defined in [TE-tunnel].

## 5. Examples of Using Different Types of YANG Models

### 5.1. Simple Connectivity Examples

The data model in [Transport-Service-Model] provides an intent-like connectivity service model which can be used in connection-oriented networks.

It would be used as follows in the ACTN architecture:

- . A CNC uses this service model to specify the two client nodes that are to be connected, and also indicates the amount of traffic (i.e., the bandwidth required) and payload type. What may be additionally specified is the SLA that describes the required quality and resilience of the service.
- . The MDSC uses the information in the request to pick the right network (domain) and also to select the provider edge nodes corresponding to the customer edge nodes.

If there are multiple domains, then the MDSC needs to coordinate across domains to set up network tunnels to deliver a service. Thus coordination includes, but is not limited to, picking the right domain sequence to deliver a service. Before it can perform such functions, it needs to get the topology information from each PNC, using topology YANG models such as [te-topology]. The topology reported from PNC to MDSC can either be abstract or non-abstract.

Additionally, an MDSC can initiate the creation of a tunnel (or tunnel segment) in order to fulfill the service request from CNC based on path computation upon the overall topology information it synthesized from different PNCs. The based model that can cater this purpose is the te-tunnel model specified in [te-tunnel].

- . Then, the PNC needs to decide the explicit route of such a tunnel or tunnel segment (in case of multiple domains), and create such a tunnel using protocols such as PCEP and RSVP-TE or using per-hop configuration.

### 5.2. VN service example

The service model defined in [ACTN-VN-YANG] describes a virtual network (VN) as a service which is a set of multiple connectivity services:

- . A CNC will request VN to the MDSC by specifying a list of VN members. Each VN member specifies either a single connectivity service, or a source with multiple potential destination points in the case that the precise destination sites are to be determined by MDSC.
  - o In the first case, the procedure is the same as the connectivity service, except that in this case, there is a list of connections requested.
  - o In the second case, where the CNC requests the MDSC to select the right destination out of a list of candidates, the MDSC needs to choose the best candidate and reply with the chosen destination for a given VN member. After this is selected, the connectivity request setup procedure is the same as in the connectivity-as-a-service example.

After the VN is set up, a successful reply message is sent from MDSC to CNC, indicating the VN is ready. This message can also be achieved by using the model defined in [ACTN-VN-YANG].

### 5.3. Data Center-Interconnection Example

This section describes more concretely how existing YANG models described in Section 4 map to an ACTN data center interconnection use case. Figure 3 shows a use-case which shows service policy-driven Data Center selection and is a reproduction of Figure A.1 from [ACTN-Info].

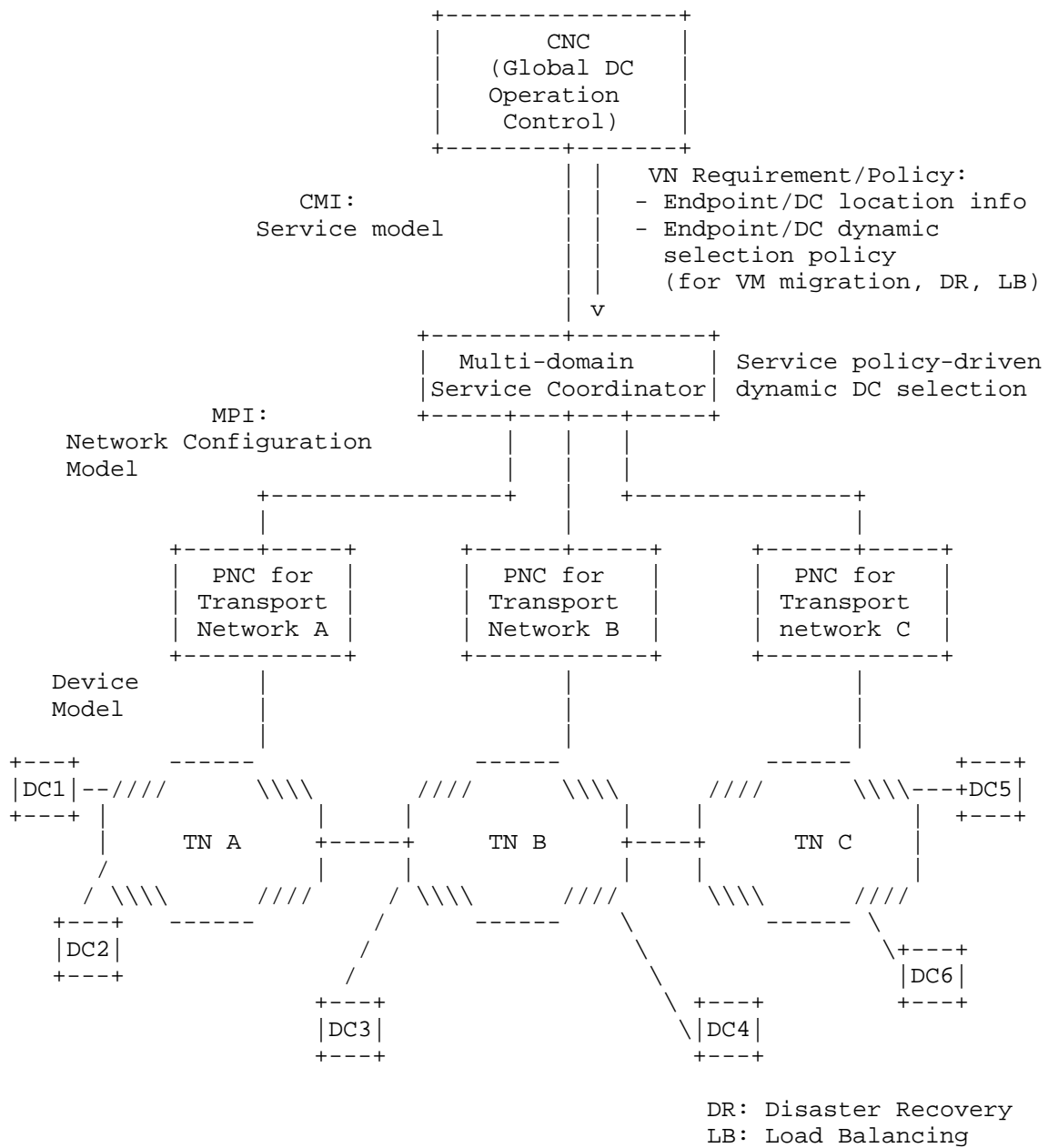


Figure 3: Service Policy-driven Data Center Selection

Figure 3 shows how VN policies from the CNC (Global Data Center Operation) are incorporated by the MDSC to support multi-destination applications. Multi-destination applications refer to applications in which the selection of the destination of a network path for a given source needs to be decided dynamically to support such applications.

Data Center selection problems arise for VM mobility, disaster recovery and load balancing cases. VN's policy plays an important role for virtual network operation. Policy can be static or dynamic. Dynamic policy for data center selection may be placed as a result of utilization of data center resources supporting VMs. The MDSC would then incorporate this information to meet the objective of this application.

#### 5.3.1. CMI (CNC-MDSC Interface)

[ACTN-VN-YANG] is used to express the definition of a VN, its VN creation request, the service objectives (metrics, QoS parameters, etc.), dynamic service policy when VM needs to be moved from one Data Center to another Data Center, etc. This service model is used between the CNC and the MDSC (CMI). The CNC in this use-case is an external entity that wants to create a VN and operates on the VN.

#### 5.3.2. MPI (MDSC-PNC Interface)

The Network Configuration Model is used between the MDSC and the PNCs. Based on the Customer Service Model's request, the MDSC will need to translate the service model into the network configuration model to instantiate a set of multi-domain connections between the prescribed sources and the destinations. The MDSC will also need to dynamically interact with the CNC for dynamic policy changes initiated by the CNC. Upon the determination of the multi-domain connections, the MDSC will need to use the network configuration model such as [TE-Tunnel] to interact with each PNC involved on the path. [TE-Topology] is used to for the purpose of underlying domain network abstraction from the PNC to the MDSC.

#### 5.3.3. PDI (PNC-Device interface)

The Device Model can be used between the PNC and its underlying devices that are controlled by the PNC. The PNC will need to trigger signaling using any mechanisms it employs (e.g. [RSVP-TE-YANG]) to provision its domain path segment. There can be a plethora of choices how to control/manage its domain network. The PNC is responsible to abstract its domain network resources and update it

to the MDSC. Note that this interface is not in the scope of ACTN. This section is provided just for an illustration purpose.

## 6. Security

This document is an informational draft. When the models mentioned in this draft are implemented, detailed security consideration will be given in such work.

How security fits into the whole architecture has the following components:

- the use of Restconf security between components
- the use of authentication and policy to govern which services can be requested by different parties.
- how security may be requested as an element of a service and mapped down to protocol security mechanisms as well as separation (slicing) of physical resources)

## 7. Acknowledgements

We thank Adrian Farrel for providing useful comments and suggestions for this draft.

## 8. References

### 8.1. Informative References

- [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.

- [ACTN-Requirements] Y. Lee, et al., "ACTN Requirements for Abstraction and Control of TE Networks", draft-ietf-teas-actn-requirements, 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.
- [ACTN-Info] Y. Lee & S. Belotti, "Information Model for Abstraction and Control of TE Networks (ACTN)", draft-leebelotti-teas-actn-info, work in progress.
- [Transport-Service-Model] X. Zhang (Editor), "A Service YANG Model for Connection-oriented Transport Networks", draft-zhang-teas-transport-service-model, work in progress.
- [PATH-COMPUTATION-API] I.Busi/S.Belotti et al. "Path Computation API", draft-busibel-ccamp-path-computation-api-00.txt, work in progress
- [RSVP-TE-YANG] T. Saad (Editor), "A YANG Data Model for Resource Reservation Protocol (RSVP)", draft-ietf-teas-yang-rsvp, work in progress.
- [Schedule] X. Liu, et. al., "A YANG Data Model for Configuration Scheduling", draft-liu-netmod-yang-schedule, work in progress.
- [ACTN-Abstraction] Y. Lee, D. Dhody, and D. Ceccarelli, "ACTN Abstraction Methods", draft-lee-tease-actn-abstraction, work in progress.
- [OTN-YANG] X. Zhang, A. Sharma, and X. Liu, "A YANG Data Model for Optical Transport Network Topology", draft-zhang-ccamp-ll-topo-yang, work in progress.

- [WSON-YANG] Y. Lee, et. al., "A Yang Data Model for WSON Optical Networks", draft-ietf-ccamp-wson-yang, work in progress.
- [Flexi-YANG] J.E. Lopez de Vergara, et. al., "YANG data model for Flexi-Grid Optical Networks", draft-vergara-ccamp-flexigrid-yang, work in progress.[ODU-Tunnel] Sharma, R. Rao, and X. Zhang, "OTN Service YANG Model", draft-sharma-ccamp-otn-service-model, work in progress.
- [ACTN-PM-Telemetry] Y. Lee, D. Dhody, S. Karunanithi, R. Vilalta, D. King, and D. Ceccarelli, "YANG models for ACTN TE Performance Monitoring Telemetry and Network Autonomics", draft-lee-teas-actn-pm-telemetry-autonomics, work in progress.
- [WSON-Tunnel] Y. Lee, D. Dhody, V. Lopez, D. King, B. Yoon, and R. Vilalta, "A Yang Data Model for WSON Tunnel", draft-lee-ccamp-wson-tunnel-model, work in progress.
- [Flexigrid-Tunnel] J. Vergara, D. Perdices, V. Lopez, O. Gonzalez de Dios, D. King, Y. Lee, and G. Galimberti, "YANG data model for Flexi-Grid media-channels", draft-vergara-ccamp-flexigrid-media-channel-yang, work in progress.

## 9. Contributors

### Contributor's Addresses

Dhruv Dhody  
Huawei Technologies

Email: dhruv.ietf@gmail.com

Xian Zhang  
Huawei Technologies

Email: zhang.xian@huawei.com



Authors' Addresses

Young Lee  
Huawei Technologies  
5340 Legacy Drive  
Plano, TX 75023, USA  
Phone: (469)277-5838

Email: leeyoung@huawei.com

Haomian Zheng  
Huawei Technologies

Email: zhenghaomian@huawei.com

Daniele Ceccarelli  
Ericsson  
Torshamnsgatan, 48  
Stockholm, Sweden

Email: daniele.ceccarelli@ericsson.com

Bin Yeong Yoon  
ETRI

Email: byyun@etri.re.kr

Oscar Gonzalez de Dios  
Telefonica

Email: oscar.gonzalezdedios@telefonica.com

Jong Yoon Shin  
SKT

Email: jongyoon.shin@sk.com

Sergio Belotti  
Nokia

Email: sergio.belotti@nokia.com



TEAS Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: September 18, 2016

Quintin Zhao  
Robin Li  
Huawei Technologies  
King Ke  
Tencent Holdings Ltd.  
Luyuan Fang  
Microsoft  
Chao Zhou  
Cisco Systems  
Boris Zhang  
Telus Communications  
March 17, 2016

The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs  
draft-zhao-teas-pcecc-use-cases-00

Abstract

In certain networks deployment scenarios, service providers would like to keep all the existing MPLS functionalities in both MPLS and GMPLS network while removing the complexity of existing signaling protocols such as LDP and RSVP-TE. In this document, we propose to use the PCE as a central controller so that LSP can be calculated/signaled/initiated/downloaded/managed through a centralized PCE server to each network devices along the LSP path while leveraging the existing PCE technologies as much as possible.

This draft describes the use cases for using the PCE as the central controller where LSPs are calculated/setup/initiated/downloaded/maintained through extending the current PCE architectures and extending the PCEP.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 18, 2016.

#### Copyright Notice

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

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

#### Table of Contents

|                                                                             |    |
|-----------------------------------------------------------------------------|----|
| 1. Introduction . . . . .                                                   | 3  |
| 1.1. Background . . . . .                                                   | 3  |
| 1.2. Using the PCE as the Central Controller (PCECC) Approach . . . . .     | 4  |
| 2. Terminology . . . . .                                                    | 7  |
| 3. PCEP Requirements . . . . .                                              | 7  |
| 4. Use Cases of PCECC for Label Resource Reservations . . . . .             | 8  |
| 5. Using PCECC for SR without the IGP Extension . . . . .                   | 9  |
| 5.1. Use Cases of PCECC for SR Best Effort(BE) Path . . . . .               | 10 |
| 5.2. Use Cases of PCECC for SR Traffic Engineering (TE) Path . . . . .      | 11 |
| 6. Use Cases of PCECC for TE LSP . . . . .                                  | 12 |
| 7. Use Cases of PCECC for Multicast LSPs . . . . .                          | 14 |
| 7.1. Using PCECC for P2MP/MP2MP LSPs' Setup . . . . .                       | 14 |
| 7.2. Use Cases of PCECC for the Resiliency of P2MP/MP2MP LSPs . . . . .     | 15 |
| 7.2.1. PCECC for the End-to-End Protection of the P2MP/MP2MP LSPs . . . . . | 15 |
| 7.2.2. PCECC for the Local Protection of the P2MP/MP2MP LSPs . . . . .      | 16 |
| 8. Use Cases of PCECC for LSP in the Network Migration . . . . .            | 17 |
| 9. Use Cases of PCECC for L3VPN and PWE3 . . . . .                          | 19 |
| 10. Using PCECC for Traffic Classification Informations . . . . .           | 19 |
| 11. The Considerations for PCECC Procedure and PCEP extensions . . . . .    | 20 |
| 12. IANA Considerations . . . . .                                           | 20 |
| 13. Security Considerations . . . . .                                       | 20 |
| 14. Acknowledgments . . . . .                                               | 20 |
| 15. References . . . . .                                                    | 20 |
| 15.1. Normative References . . . . .                                        | 20 |
| 15.2. Informative References . . . . .                                      | 21 |
| Authors' Addresses . . . . .                                                | 22 |

## 1. Introduction

### 1.1. Background

In certain network deployment scenarios, service providers would like to have the ability to dynamically adapt to a wide range of customer's requests for the sake of flexible network service delivery, SDN has provides additional flexibility in how the network is operated comparing the traditional network.

The existing networking ecosystem has become awfully complex and highly demanding in terms of robustness, performance, scalability, flexibility, agility, etc. By migrating to the SDN enabled network from the existing network, service providers and network operators must have a solution which they can evolve easily from the existing network into the SDN enabled network while keeping the network services remain scalable, guarantee robustness and availability etc.

Taking the smooth transition between traditional network and the new SDN enabled network into account, especially from a cost impact assessment perspective, using the existing PCE components from the current network to function as the central controller of the SDN network is one choice, which not only achieves the goal of having a centralized controller to provide the functionalities needed for the central controller, but also leverages the existing PCE network components.

The Path Computation Element communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform route computations in response to Path Computation Clients (PCCs) requests. PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model draft [I-D. draft-ietf-pce- stateful-pce] describes a set of extensions to PCEP to enable active control of MPLS-TE and GMPLS tunnels.

[I-D.crabbe-pce-pce-initiated-lsp] describes the setup and teardown of PCE-initiated LSPs under the active stateful PCE model, without the need for local configuration on the PCC, thus allowing for a dynamic MPLS network that is centrally controlled and deployed.

[I-D.ali-pce-remote-initiated-gmpls-lsp] complements [I-D. draft-crabbe-pce-pce-initiated-lsp] by addressing the requirements for remote-initiated GMPLS LSPs.

SR technology leverages the source routing and tunneling paradigms. A source node can choose a path without relying on hop-by-hop signaling protocols such as LDP or RSVP-TE. Each path is specified as a set of "segments" advertised by link-state routing protocols

(IS-IS or OSPF). [I-D.filsfils-spring-segment-routing] provides an introduction to SR technology. The corresponding IS-IS and OSPF extensions are specified in [I-D.ietf-isis-segment-routing-extensions] and [I-D.psenak-ospf-segment-routing-extensions], respectively.

A Segment Routed path (SR path) can be derived from an IGP Shortest Path Tree (SPT). Segment Routed Traffic Engineering paths (SR-TE paths) may not follow IGP SPT. Such paths may be chosen by a suitable network planning tool and provisioned on the source node of the SR-TE path.

It is possible to use a stateful PCE for computing one or more SR-TE paths taking into account various constraints and objective functions. Once a path is chosen, the stateful PCE can instantiate an SR-TE path on a PCC using PCEP extensions specified in [I-D.crabbe-pce-pce-initiated-lsp] using the SR specific PCEP extensions described in [I-D.sivabalan-pce-segment-routing].

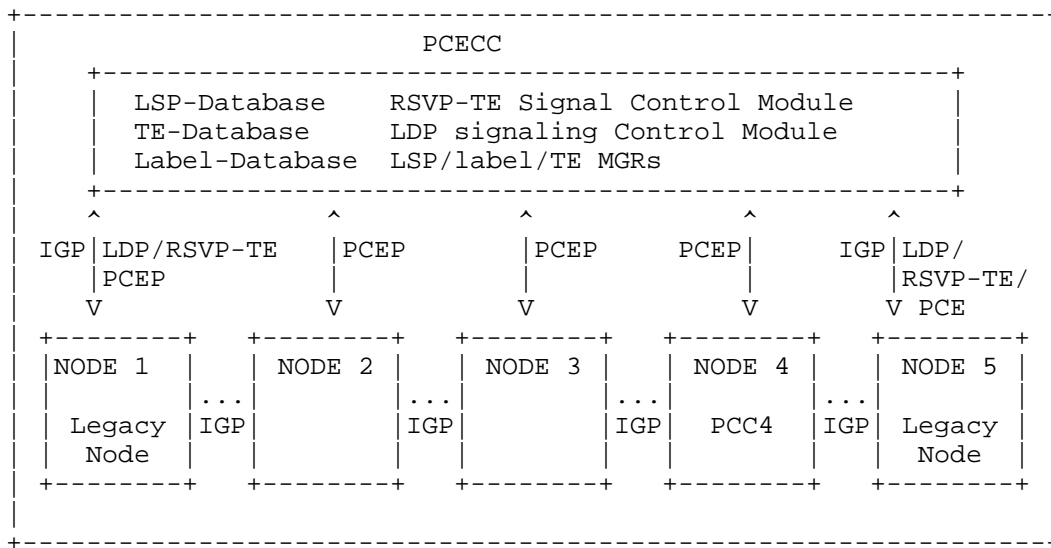
By using the solutions provided from above drafts, LSP in both MPLS and GMPLS network can be setup/delete/maintained/synchronized through a centrally controlled dynamic MPLS network. Since in these solutions, the LSP is need to be signaled through the head end LER to the tail end LER, there are either RSVP-TE signaling protocol need to be deployed in the MPLS/GMPLS network, or extend TGP protocol with node/adjacency segment identifiers signaling capability to be deployed.

The PCECC solution proposed in this document allow for a dynamic MPLS network that is eventually controlled and deployed without the deployment of RSVP-TE protocol or extended IGP protocol with node/adjacency segment identifiers signaling capability while providing all the key MPLS functionalities needed by the service providers. These key MPLS features include MPLS P2P LSP, P2MP/MP2MP LSP, MPLS protection mechanism etc. In the case that one LSP path consists legacy network nodes and the new network nodes which are centrally controlled, the PCECC solution provides a smooth transition step for users.

#### 1.2. Using the PCE as the Central Controller (PCECC) Approach

With PCECC, it not only removes the existing MPLS signaling totally from the control plane without losing any existing MPLS functionalities, but also PCECC achieves this goal through utilizing the existing PCEP without introducing a new protocol into the network.

The following diagram illustrates the PCECC architecture.



Through the draft, we call the combination of the functionality for global label range signaling and the functionality of LSP setup/download/cleanup using the combination of global labels and local labels as PCECC functionality.

Current MPLS label has local meaning. That is, MPLS label allocated locally and signaled through the LDP/RSVP-TE/BGP etc dynamic signaling protocol.

As the SDN(Service-Driven Network) technology develops, MPLS global label has been proposed again for new solutions. [I-D.li-mpls-global-label-usecases] proposes possible usecases of MPLS global label. MPLS global label can be used for identification of the location, the service and the network in different application scenarios. From these usecases we can see that no matter SDN or traditional application scenarios, the new solutions based on MPLS global label can gain advantage over the existing solutions to facilitate service provisions. The solution choices are described in [I-D.li-mpls-global-label-framework].

To ease the label allocation and signaling mechanism, also with the new applications such as concentrated LSP controller is introduced, PCE can be conveniently used as a central controller and MPLS global label range negotiator.

The later section of this draft describes the user cases for PCE server and PCE clients to have the global label range negotiation and local label range negotiation functionality.

To empower networking with centralized controllable modules, there are many choices for downloading the forwarding entries to the data plane, one way is the use of the OpenFlow protocol, which helps devices populate their forwarding tables according to a set of instructions to the data plane. There are other candidate protocols to convey specific configuration information towards devices also. Since the PCEP protocol is already deployed in some of the service network, to leverage the PCEP to populated the MPLS forwarding table is a possible good choice.

For the centralized network, the performance achieved through distributed system can not be easy matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that the adjacency IDs for all the network nodes and links are propagated through the centralized controller instead of using the IGP extension.

The node and link adjacency IDs can be negotiated through the PCECC with each PCECC clients and these IDs can be just taken from the global label range which has been negotiated already.

With the capability of supporting SR within the PCECC architecture, all the p2p forwarding path protection use cases described in the draft [I-D.ietf-spring-resiliency-use-cases] will be supported too within the PCECC network. These protection alternatives include end-to-end path protection, local protection without operator management and local protection with operator management.

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP lsp using the local label range for each network nodes.

With the capability of setting up/maintaining the P2MP/MP2MP LSP within the PCECC network, it is easy to provide the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.



## 2. Terminology

The following terminology is used in this document.

IGP: Interior Gateway Protocol. Either of the two routing protocols, Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS).

PCC: Path Computation Client: any client application requesting a path computation to be performed by a Path Computation Element.

PCE: Path Computation Element. An entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

TE: Traffic Engineering.

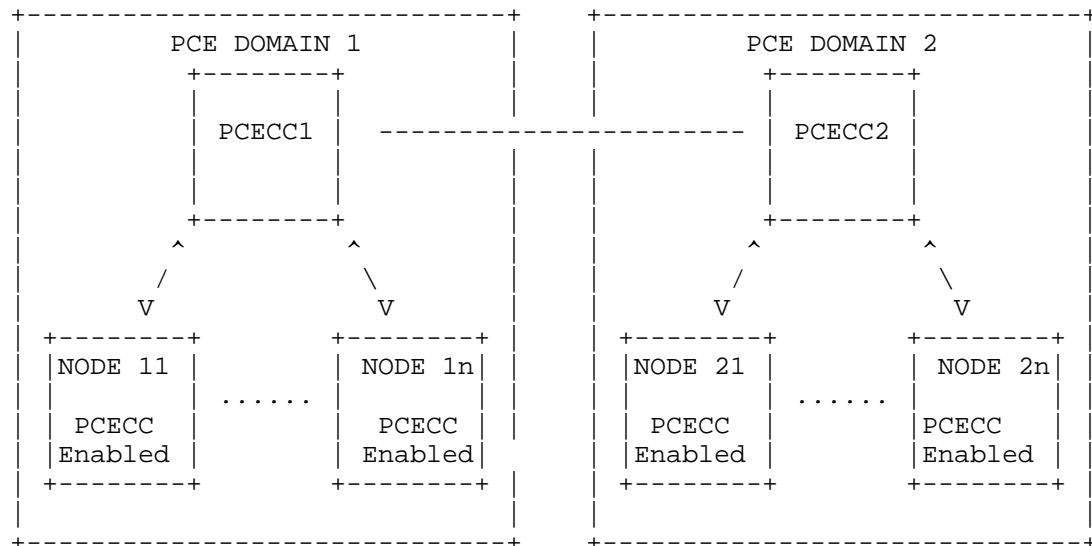
## 3. PCEP Requirements

Following key requirements associated PCECC should be considered when designing the PCECC based solution:

1. Path Computation Element (PCE) clients supporting this draft MUST have the capability to advertise its PCECC capability to the PCECC.
2. Path Computation Element (PCE) supporting this draft MUST have the capability to negotiate a global label range for a group of clients.
3. Path Computation Client (PCC) MUST be able ask for global label range assigned in path request message .
4. PCE are not required to support label reserve service. Therefore, it MUST be possible for a PCE to reject a Path Computation Request message with a reason code that indicates no support for label reserve service.
5. PCEP SHOULD provide a means to return global label range and LSP label assignments of the computed path in the reply message.
6. PCEP SHOULD provide a means to download the MPLS forwarding entry to the PCECC's clients.

#### 4. Use Cases of PCECC for Label Resource Reservations

Example 1 to 2 are based on network configurations illustrated using the following figure:



##### Example 1: Shared Global Label Range Reservation

- o PCECC Clients nodes report MPLS label capability to the central controller PCECC.
- o The central controller PCECC collects MPLS label capability of all nodes. Then PCECC can calculate the shared MPLS global label range for all the PCECC client nodes.
- o In the case that the shared global label range need to be negotiated across multiple domains, the central controllers of these domains need to be communicate to negotiate a common global label range.
- o The central controller PCECC notifies the shared global label range to all PCECC client nodes.

##### Example 2: Global Label Allocation

- o PCECC Client node1 send global label allocation request to the central controller PCECC1.

- o The central controller PCECC1 allocates the global label for FEC1 from the shared global label range and sends the reply to the client node1.
- o The central controller PCECC1 notifies the allocated label for FEC1 to all PCECC client nodes within domain 1.

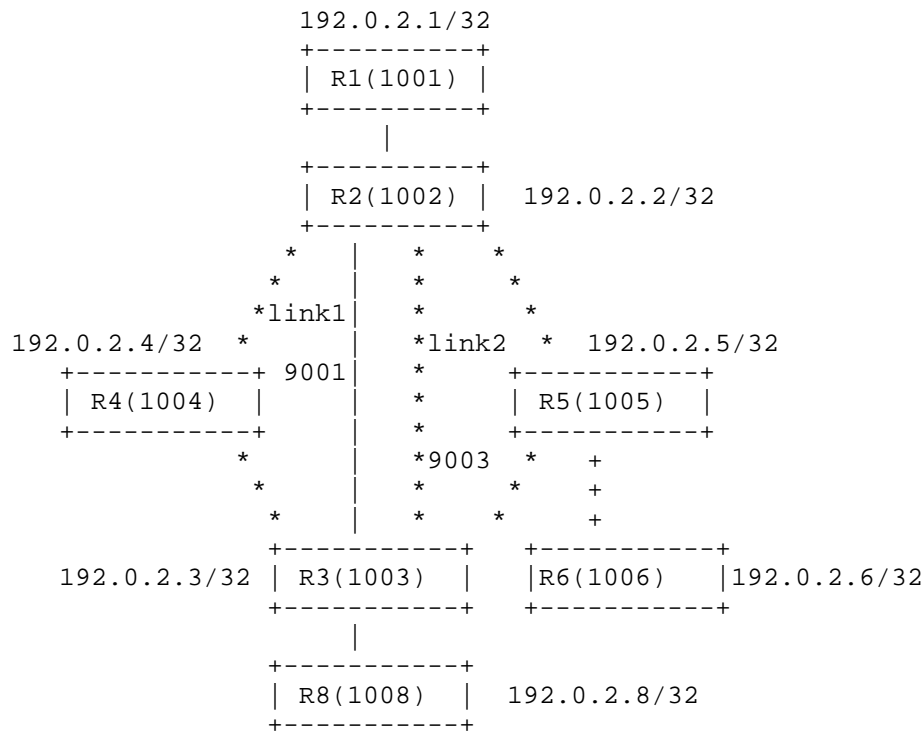
## 5. Using PCECC for SR without the IGP Extension

For the centralized network, the performance achieved through distributed system can not be easily matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that node segment IDs and adjacency segment IDs for all the network are allocated dynamically and propagated through the centralized controller instead of using the IGP extension.

When the PCECC is used for the distribution of the node segment ID and adjacency segment ID, the node segment ID is allocated from the global label pool. For the allocation of adjacency segment ID, there are two choices, the first choice is that it is allocated from the local label pool, the second choice is that it is allocated from the global label pool. The advantage for the second choice is that the depth of the label stack for the forwarding path encoding will be reduced since adjacency segment ID can signal the forwarding path without adding the node segment ID in front of it. In this version of the draft, we use the first choice for now. We may update the draft to reflect the use of the second choice.

Same as the SR solutions, when PCECC is used as the central controller, the support of FRR on any topology can be pre-computed and setup without any additional signaling (other than the regular IGP/BGP protocols) including the support of shared risk constraints, support of node and link protection and support of microloop avoidance.

The following example illustrates the use case where the node segment ID and adjacency segment ID are allocated from the global label allocated for SR path.



### 5.1. Use Cases of PCECC for SR Best Effort(BE) Path

In this mode of the solution, the PCECC just need to allocate the node segment ID and adjacency ID without calculating the explicit path for the SR path. The ingress of the forwarding path just need to encapsulate the destination node segment ID on top of the packet. All the intermediate nodes will forward the packet based on the final destination node segment id. It is similar to the LDP LSP forwarding except that label swapping is using the same global label both for the in segment and out segment in each hop.

The p2p SR BE path examples are explained as bellow:

Note that the node segment id for each node from the shared global labels ranges negotiated already.

Example 1:

R1 may send a packet to R8 simply by pushing an SR header with segment list {1008}. The path can be: R1-R2-R3-R8 or R1-R2-R5-R8 depending on the route calculation on node R2.

#### Example 2: local link/node protection:

For the packet which has destination of R3 and after that, R2 may preinstalled the backup forwarding entry to protect the R4 node, the pre-installed the backup path can go through either node5 or link1 or link2 between R2 and R3. The backup path calculation is locally decided by R2 and any existing IP FRR algorithms can be used here.

### 5.2. Use Cases of PCECC for SR Traffic Engineering (TE) Path

In the case of traffic engineering path is needed, the PCECC need to allocate the node segment ID and adjacency ID, and at the same time PCECC calculates the explicit path for the SR path and pass this explicit path represented with a sequence of node segment id and adjacency id. The ingress of the forwarding path need to encapsulate the stack of node segment id and adjacency id on top of the packet. For the case where strict traffic engineering path is needed, all the intermediate nodes and links will be specified through the stack of labels so that the packet is forwarded exactly as it is wanted.

Even though it is similar to TE LSP forwarding where forwarding path is engineered, but the Qos is only guaranteed through the enforce of the bandwidth admission control. As for the RSVP-TE LSP case, Qos is guaranteed through the link bandwidth reservation in each hop of the forwarding path.

The p2p SR traffic engineering path examples are explained as bellow:

Note that the node segment id for each node is allocated from the shared global labels ranges negotiated already and adjacency segment ids for each link are allocated from the local label pool for each node.

#### Example 1:

R1 may send a packet P1 to R8 simply by pushing an SR header with segment list {1008}. The path should be: R1-R2-R3-R8.

#### Example 2:

R1 may send a packet P2 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.

#### Example 3:

R1 may send a packet P3 to R8 while avoiding the links between R2 and R3 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8

The p2p local protection examples for SR TE path are explained as below:

Example 4: local link protection:

- o R1 may send a packet P4 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.
- o When node R2 receives the packet from R1 which has the header of R2- (1)link-R3-R8, and also find out there is a link failure of link1, then it will send out the packet with header of R3-R8 through link2.

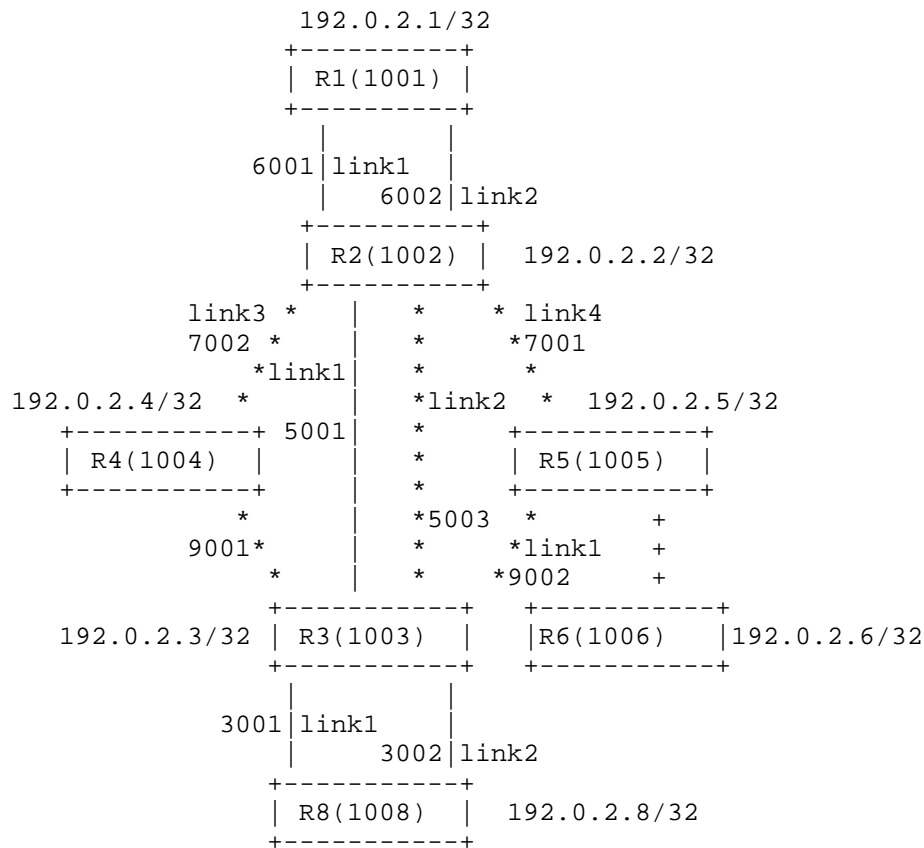
Example 5: local node protection:

- o R1 may send a packet P5 to R8 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8.
- o When node R2 receives the packet from R1 which has the header of {1004, 1008}, and also find out there is a node failure for node4, then it will send out the packet with header of {1005, 1008} to node5 instead of node4.

## 6. Use Cases of PCECC for TE LSP

In the previous sections, we have discussed the cases where the SR path is setup through the PCECC. Although those cases give the simplicity and scalability, but there are existing functionalities for the traffic engineering path such as the bandwidth guarantee through the full forwarding path and the multicast forwarding path which SR based solution cannot solve. Also there are cases where the depth of the label stack may have been an issue for existing deployment and certain vendors.

So to address these issues, PCECC architecture should also support the TE LSP and multicast LSP functionalities. To achieve this, the existing PCEP can be used to communicate between the PCE server and PCE's client PCC for exchanging the path request and reply information regarding to the TE LSP info. In this case, the TE LSP info is not only the path info itself, but it includes the full forwarding info. Instead of letting the ingress of LSP to initiate the LSP setup through the RSVP-TE signaling protocol, with minor extensions, we can use the PCEP to download the complete TE LSP forwarding entries for each node in the network.



#### TE LSP Setup Example

- o Node1 sends a path request message for the setup of TE LSP from R1 to R8.
- o PCECC program each node along the path from R1 to R8 with the primary path: {R1, link1, 6001}, {R2, link3, 7002}, {R4, link0, 9001}, {R3, link1, 3001}, {R8}.
- o For the end to end protection, PCECC program each node along the path from R1 to R8 with the secondary path: {R1, link2, 6002}, {R2, link4, 7001}, {R5, link1, 9002}, {R3, link2, 3002}, {R8}.
- o It is also possible to have a secondary backup path for the local node protection setup by PCECC. For example, the primary path is still same as what we have setup so far, then to protect the node R4 locally, PCECC can program the secondary path like this: {R1,

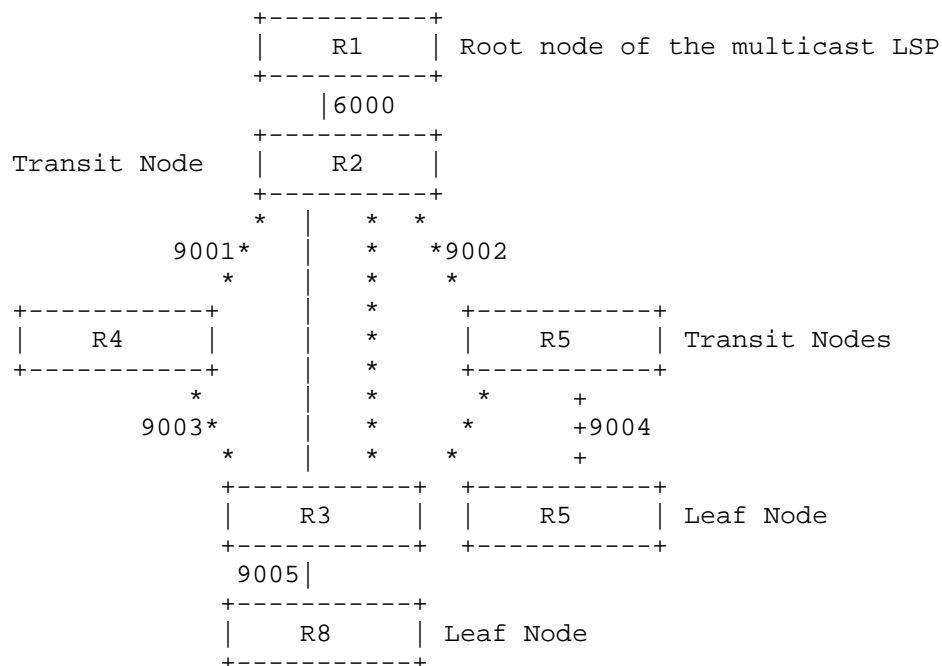
link1, 6001}, {R2, link1, 5001}, {R3, link1, 3001}, {R8}. By doing this, the node R4 is locally protected.

## 7. Use Cases of PCECC for Multicast LSPs

The current multicast LSPs are setup either using the RSVP-TE P2MP or mLDP protocols. The setup of these LSPs not only need a lot of manual configurations, but also it is also complex when the protection is considered. By using the PCECC solution, the multicast LSP can be computed and setup through centralized controller which has the full picture of the topology and bandwidth usage for each link. It not only reduces the complex configurations comparing the distributed RSVP-TE P2MP or mLDP signal lings, but also it can compute the disjoint primary path and secondary path efficiently.

### 7.1. Using PCECC for P2MP/MP2MP LSPs' Setup

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP lsp using the local label range for each network nodes.





The P2MP examples are explained here:

Step1: R1 may send a packet P1 to R2 simply by pushing an label of 6000 to the packet.

Step2: After R2 receives the packet with label 6000, it will forwarding to R4 by pushing header of 9001 and R5 by pusing header of 9002.

Step3: After R4 receives the packet with label 9001, it will forwarding to R3 by pushing header of 9003. After R5 receives the packet with label 9002, it will forwarding to R5 by pushing header of 9004.

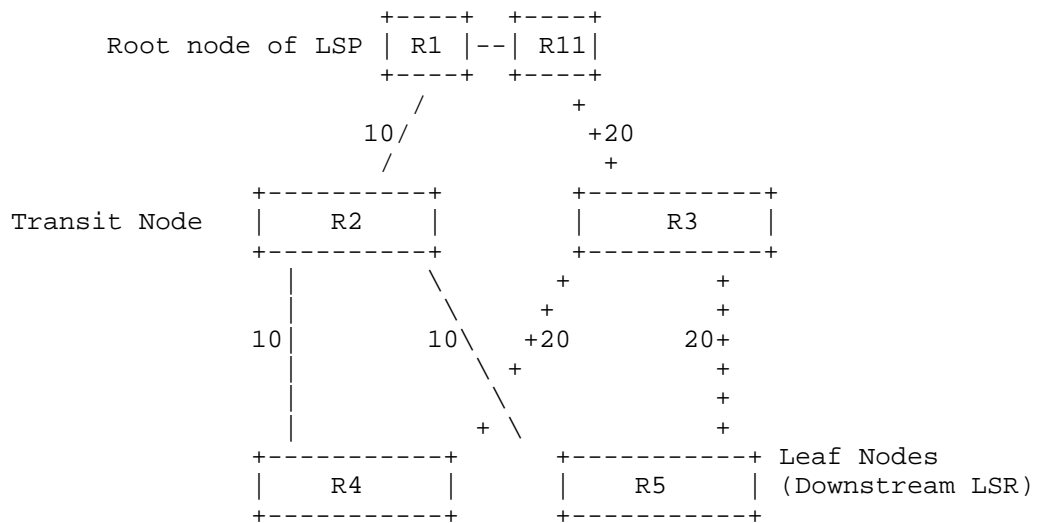
Step3: After R3 receives the packet with label 9003, it will forwarding to R8 by pushing header of 9005

## 7.2. Use Cases of PCECC for the Resiliency of P2MP/MP2MP LSPs

### 7.2.1. PCECC for the End-to-End Protection of the P2MP/MP2MP LSPs

In this section we describe the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.

An end-to-end protection (for nodes and links) principle can be applied for computing backup P2MP or MP2MP LSPs. During computation of the primarily multicast trees, PCECC server may also be taken into consideration to compute a secondary tree. A PCE may compute the primary and backup P2MP or MP2Mp LSP together or sequentially.

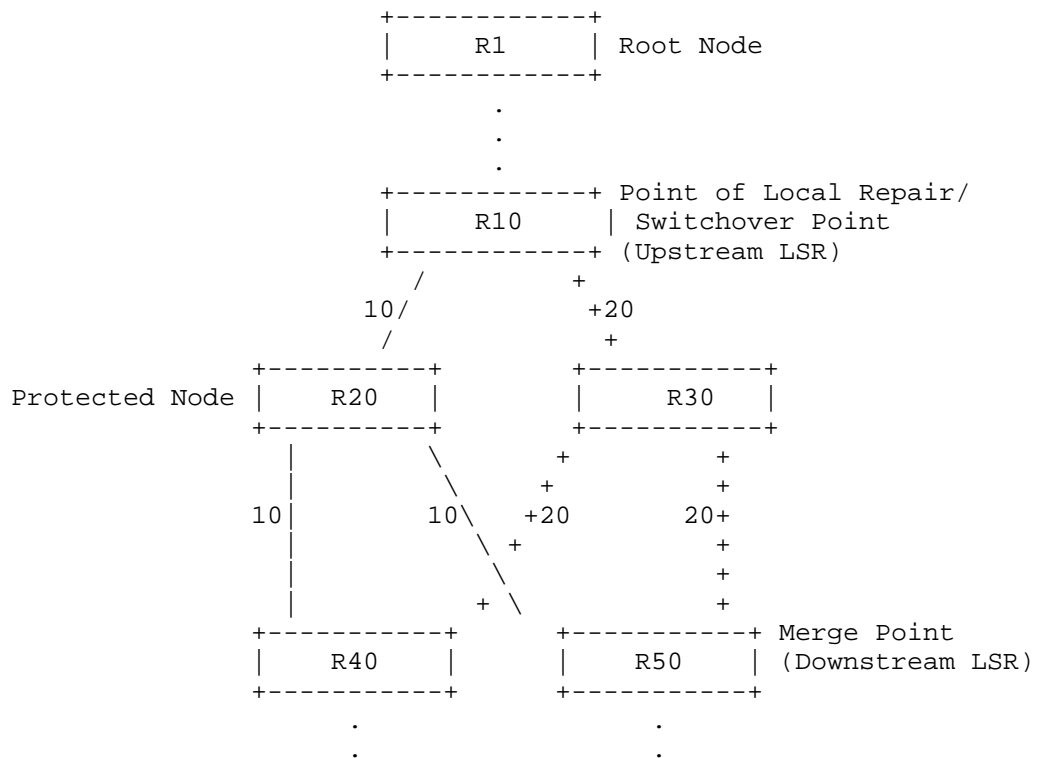


In the example above, when the PCECC setup the primary multicast tree from the root node R1 to the leafs, which is R1->R2->{R4, R5}, at same time, it can setup the backup tree, which is R11->R3->{R4, R5}. Both the these two primary forwarding tree and secondary forwarding tree will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R1->R2->{R4, R5} path normally, and when there is a node in the primary tree, then the root node R1 will switch the flow to the backup tree, which is R11->R3->{R4, R5}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

#### 7.2.2. PCECC for the Local Protection of the P2MP/MP2MP LSPs

In this section we describe the local protection service in the PCECC network for the P2MP/MP2MP LSP.

While the PCECC sets up the primary multicast tree, it can also build the back LSP among PLR, the protected node, and MPs (the downstream nodes of the protected node). In the cases where the amount of downstream nodes are huge, this mechanism can avoid unnecessary packet duplication on PLR, so that protect the network from traffic congestion risk.



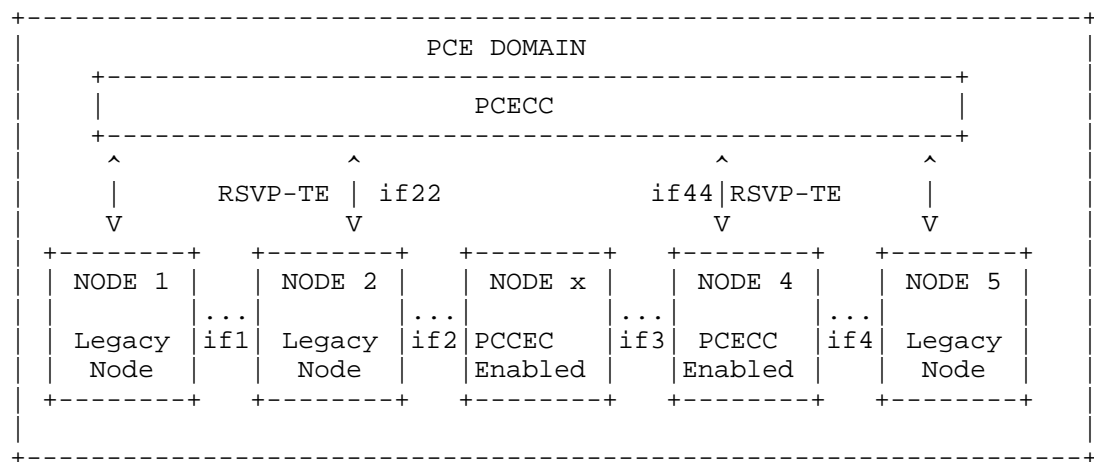
In the example above, when the PCECC setup the primary multicast path around the PLR node R10 to protect node R20, which is R10->R20->{R40, R50}, at same time, it can setup the backup path R10->R30->{R40, R50}. Both the these two primary forwarding path and secondary forwarding path will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R10->R20->{R40, R50} path normally, and when there is a node failure for node R20, then the PLR node R10 will switch the flow to the backup path, which is R10->R30->{R40, R50}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

#### 8. Use Cases of PCECC for LSP in the Network Migration

One of the main advantages for PCECC solution is that it has backward compatibility naturally since the PCE server itself can function as a proxy node of MPLS network for all the new nodes which don't support the existing MPLS signaling protocol anymore.

As it is illustrated in the following example, the current network will migrate to a total PCECC controlled network gradually by replacing the legacy nodes. During the migration, the legacy nodes still need to signal using the existing MPLS protocol such as LDP and RSVP-TE, and the new nodes setup their portion of the forwarding path through PCECC directly. With the PCECC function as the proxy of these new nodes, MPLS signaling can populate through network as normal.

Example described in this section is based on network configurations illustrated using the following figure:



#### Example: PCECC Initiated LSP Setup In the Network Migration

In this example, there are five nodes for the TE LSP from head end (node1) to the tail end (node5). Where the NodeX is central controlled and other nodes are legacy nodes.

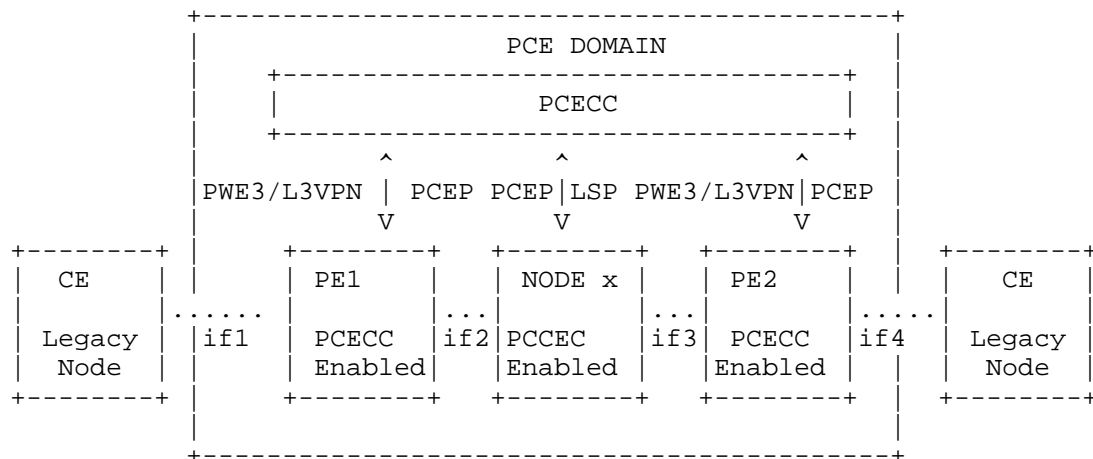
- o Node1 sends a path request message for the setup of LSP destinating to Node5.
- o PCECC sends a reply message for LSP setup with path (node1, if1), (node2, if22), (node-PCECC, if44), (node4, if4), Nnode5.
- o Node1, Node2, Node-PCECC, Node 5 will setup the LSP to Node5 normally using the local label as normal.
- o Then the PCECC will program the outsegment of Node2, the insegment of Node4, and the insegment/outsegment for NodeX.

## 9. Use Cases of PCECC for L3VPN and PWE3

The existing services using MPLS LSP tunnels based on MPLS signalling mechanism such L3VPN, PWE3 and IPv6 can be simplified by using the PCECC to negotiate the label assignments for the L3VPN, PWE3 and Ipv6.

In the case of L3VPN, VPN labels can be negotiated and distributed through the PCECC PCEP among the PE router instead of using the BGP protocols.

Example described in this section is based on network configurations illustrated using the following figure:



Example: Using PCECC for L3VPN and PWE3

In the case of PWE3, instead of using the LDP signalling protocols, the label and port pairs assigned to each pseudowire can be negotiated through PCECC among the PE routers and the corresponding forwarding entries will be distributed into each PE routers through the extended PCEP protocols.

## 10. Using PCECC for Traffic Classification Informations

When a TE-LSP is set up, the head end needs to know:

- o how to use it

- o What traffic to send on the LSp
- o Whether it is a virtual link
- o Whether to advertise it in the IGP
- o What bits of this information to signal to the tail end

PCEP allows an Active PCE to set up or modify LSPs. But we have no way to tell the head end how to use the LSP This is because of history. It used to be the LER that made the request of the PCE, so it knew why it wanted the LSP.

With the PCECC architecture by extending the PCEP protocols, is is easy to carry these informations such as how to use the LSP, how to advertise the LSP and other extra signaling information.

#### 11. The Considerations for PCECC Procedure and PCEP extensions

The PCECC's procedures and PCEP extensions is defined in [I-D.zhao-pce-pcep-extension-for-pce-controller].

#### 12. IANA Considerations

This document does not require any action from IANA.

#### 13. Security Considerations

TBD.

#### 14. Acknowledgments

We would like to thank Robert Tao, Changjiang Yan, Tieying Huang and Adrian Farrel for their useful comments and suggestions.

#### 15. References

##### 15.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<http://www.rfc-editor.org/info/rfc5440>>.

## 15.2. Informative References

- [RFC5441] Vasseur, JP., Ed., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, DOI 10.17487/RFC5441, April 2009, <<http://www.rfc-editor.org/info/rfc5441>>.
- [RFC5541] Le Roux, JL., Vasseur, JP., and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, DOI 10.17487/RFC5541, June 2009, <<http://www.rfc-editor.org/info/rfc5541>>.
- [I-D.filsfils-spring-segment-routing]  
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.
- [I-D.ietf-pce-stateful-pce]  
Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-13 (work in progress), December 2015.
- [I-D.crabbe-pce-pce-initiated-lsp]  
Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-crabbe-pce-pce-initiated-lsp-03 (work in progress), October 2013.
- [I-D.ali-pce-remote-initiated-gmpls-lsp]  
Ali, Z., Sivabalan, S., Filsfils, C., Varga, R., Lopez, V., Dios, O., and X. Zhang, "Path Computation Element Communication Protocol (PCEP) Extensions for remote-initiated GMPLS LSP Setup", draft-ali-pce-remote-initiated-gmpls-lsp-03 (work in progress), February 2014.
- [I-D.ietf-isis-segment-routing-extensions]  
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-06 (work in progress), December 2015.

- [I-D.psenak-ospf-segment-routing-extensions]  
Psenak, P., Previdi, S., Filsfils, C., Gredler, H.,  
Shakir, R., Henderickx, W., and J. Tantsura, "OSPF  
Extensions for Segment Routing", draft-psenak-ospf-  
segment-routing-extensions-05 (work in progress), June  
2014.
- [I-D.sivabalan-pce-segment-routing]  
Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E.,  
Raszuk, R., Lopez, V., and J. Tantsura, "PCEP Extensions  
for Segment Routing", draft-sivabalan-pce-segment-  
routing-03 (work in progress), July 2014.
- [I-D.li-mpls-global-label-usecases]  
Li, Z., Zhao, Q., Yang, T., Raszuk, R., and L. Fang,  
"Usecases of MPLS Global Label", draft-li-mpls-global-  
label-usecases-03 (work in progress), October 2015.
- [I-D.li-mpls-global-label-framework]  
Li, Z., Zhao, Q., Chen, X., Yang, T., and R. Raszuk, "A  
Framework of MPLS Global Label", draft-li-mpls-global-  
label-framework-02 (work in progress), July 2014.
- [I-D.zhao-pce-pcep-extension-for-pce-controller]  
Zhao, Q., Li, Z., Dhody, D., and C. Zhou, "PCEP Procedures  
and Protocol Extensions for Using PCE as a Central  
Controller (PCECC) of LSPs", draft-zhao-pce-pcep-  
extension-for-pce-controller-03 (work in progress), March  
2016.
- [I-D.ietf-spring-resiliency-use-cases]  
Francois, P., Filsfils, C., Decraene, B., and R. Shakir,  
"Use-cases for Resiliency in SPRING", draft-ietf-spring-  
resiliency-use-cases-02 (work in progress), December 2015.

#### Authors' Addresses

Quintin Zhao  
Huawei Technologies  
125 Nagog Technology Park  
Acton, MA 01719  
US  
  
EMail: quintin.zhao@huawei.com



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

EMail: lizhenbin@huawei.com

King Ke  
Tencent Holdings Ltd.  
Shenzhen  
China

EMail: kinghe@tencent.com

Luyuan Fang  
Microsoft

EMail: lufang@microsoft.com

Chao Zhou  
Cisco Systems

EMail: chao.zhou@cisco.com

Boris Zhang  
Telus Communications

EMail: Boris.zhang@telus.com

TEAS Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: May 25, 2017

Quintin Zhao  
Robin Li  
Boris Khasanov, Ed.  
Huawei Technologies  
King Ke  
Tencent Holdings Ltd.  
Luyuan Fang  
Microsoft  
Chao Zhou  
Cisco Systems  
Boris Zhang  
Telus Communications  
Artem Rachitskiy  
Anton Gulida  
Mobile TeleSystems JLLC  
October 26, 2016

The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs  
draft-zhao-teas-pcecc-use-cases-02

Abstract

In certain networks deployment scenarios, service providers would like to keep all the existing MPLS functionalities in both MPLS and GMPLS network while reducing existing complexity. In this document, we propose to use the PCE as a central controller so that LSP can be calculated/signaled/initiated/downloaded/managed through a centralized PCE server to each network devices along the LSP path while leveraging the existing PCE technologies as much as possible.

This draft describes the use cases for using the PCE as the central controller where LSPs are calculated/setup/initiated/downloaded/maintained through extending the current PCE architectures and extending the PCEP.

Requirements Language

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

Status of This Memo

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

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

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



This Internet-Draft will expire on May 25, 2017.

#### Copyright Notice

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

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

#### Table of Contents

|                                                                                                              |    |
|--------------------------------------------------------------------------------------------------------------|----|
| 1. Introduction . . . . .                                                                                    | 3  |
| 1.1. Background . . . . .                                                                                    | 3  |
| 1.2. Using the PCE as the Central Controller (PCECC) Approach . . . . .                                      | 4  |
| 2. Terminology . . . . .                                                                                     | 7  |
| 3. PCEP Requirements . . . . .                                                                               | 7  |
| 4. Use Cases of PCECC for Label Resource Reservations . . . . .                                              | 8  |
| 5. Using PCECC for SR without the IGP Extension . . . . .                                                    | 9  |
| 5.1. Use Cases of PCECC for SR Best Effort(BE) Path . . . . .                                                | 10 |
| 5.2. Use Cases of PCECC for SR Traffic Engineering (TE) Path . . . . .                                       | 11 |
| 6. Use Cases of PCECC for TE LSP . . . . .                                                                   | 12 |
| 7. Use Cases of PCECC for Multicast LSPs . . . . .                                                           | 14 |
| 7.1. Using PCECC for P2MP/MP2MP LSPs' Setup . . . . .                                                        | 14 |
| 7.2. Use Cases of PCECC for the Resiliency of P2MP/MP2MP LSPs . . . . .                                      | 15 |
| 7.2.1. PCECC for the End-to-End Protection of the P2MP/MP2MP LSPs . . . . .                                  | 15 |
| 7.2.2. PCECC for the Local Protection of the P2MP/MP2MP LSPs . . . . .                                       | 16 |
| 8. Use Cases of PCECC for LSP in the Network Migration . . . . .                                             | 17 |
| 9. Use Cases of PCECC for L3VPN and PWE3 . . . . .                                                           | 19 |
| 10. Using PCECC for Traffic Classification Informations . . . . .                                            | 19 |
| 11. Use case of PCECC for load balancing . . . . .                                                           | 20 |
| 12. Using reliable P2MP TE based multicast delivery for distributed computations (MapReduce-Hadoop). . . . . | 22 |
| 13. PCECC and Inter-AS TE . . . . .                                                                          | 24 |
| 14. The Considerations for PCECC Procedure and PCEP extensions . . . . .                                     | 25 |
| 14. IANA Considerations . . . . .                                                                            | 25 |
| 15. Security Considerations . . . . .                                                                        | 25 |
| 16. Acknowledgments . . . . .                                                                                | 25 |
| 17. References . . . . .                                                                                     | 25 |
| 17.1. Normative References . . . . .                                                                         | 25 |
| 17.2. Informative References . . . . .                                                                       | 25 |
| Authors' Addresses . . . . .                                                                                 | 26 |

## 1. Introduction

### 1.1. Background

In many network deployment scenarios, service providers would like to have the ability to dynamically adapt to a wide range of customer's requests for the sake of flexible network service delivery. SDN provides such flexibility and programmability for that case.

By migrating to the SDN enabled network from the existing network, service providers and network operators must have a solution which they can easily evolve from the existing network into the fully SDN enabled network while keeping scalability of the network services, guarantee robustness, availability, flexibility etc.

Taking into account the smooth transition from existing network to the new SDN enabled network with optimal cost, re-usage of the existing PCE components in network to be function of the central (SDN) controller is one choice, that not only achieves the goal of having centralized control but also leverages the existing PCE network components.

The Path Computation Element communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform route computations in response to Path Computation Clients (PCCs) requests. PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model draft [I-D. draft-ietf-pce-stateful-pce] describes a set of extensions to PCEP to enable active control of MPLS-TE and GMPLS tunnels.

[I-D.crabbe-pce-pce-initiated-lsp] describes the setup and teardown of PCE-initiated LSPs under the active stateful PCE model, without the need for local configuration on the PCC, thus allowing for a dynamic MPLS network that is centrally controlled and deployed.

[I-D.ali-pce-remote-initiated-gmpls-lsp] complements [I-D. draft-crabbe-pce-pce-initiated-lsp] by addressing the requirements for remote-initiated GMPLS LSPs.

Segment Routing (SR) technology leverages the source routing and tunneling paradigms. A source node can choose a path without relying on hop-by-hop signaling protocols such as LDP or RSVP-TE. Each path is specified as a set of "segments" advertised by link-state routing

protocols (IS-IS or OSPF). [I-D.filsfils-spring-segment-routing] provides an introduction to SR technology. The corresponding IS-IS and OSPF extensions are specified in [I-D.ietf-isis-segment-routing-extensions] and [I-D.psenak-ospf-segment-routing-extensions], respectively.

A Segment Routed path (SR path) can be derived from an IGP Shortest Path Tree (SPT). Segment Routed Traffic Engineering paths (SR-TE paths) may not follow IGP SPT. Such paths may be chosen by a suitable network planning tool and provisioned on the source node of the SR-TE path.

It is possible to use a stateful PCE for computing one or more SR-TE paths taking into account various constraints and objective functions. Once a path is chosen, the stateful PCE can instantiate an SR-TE path on a PCC using PCEP extensions specified in [I-D.crabbe-pce-pce-initiated-lsp] using the SR specific PCEP extensions described in [I-D.sivabalan-pce-segment-routing].

By using the solutions provided from above drafts, LSP in both MPLS and GMPLS network can be setup/delete/maintained/synchronized through a centrally controlled MPLS network.

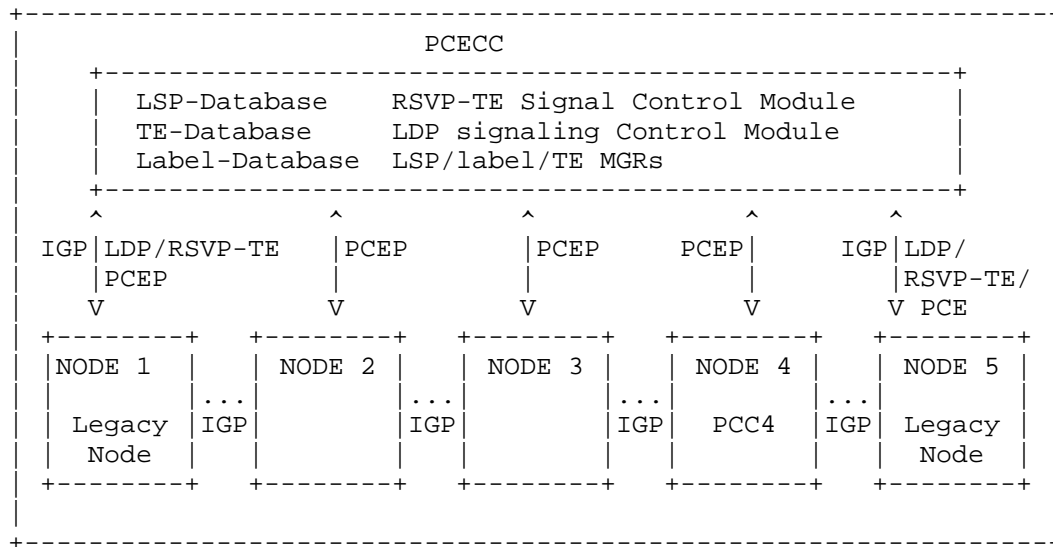
The PCECC solution proposed in this document allows creation of dynamic MPLS network that is eventually controlled and deployed without the RSVP-TE protocol or extended IGP protocol with node/adjacency segment identifiers while providing all the key MPLS functionalities needed by the service providers.

These key MPLS features include MPLS P2P LSP, P2MP/MP2MP LSP, MPLS protection mechanism etc. In the case that one LSP path consists legacy network nodes and the new network nodes which are centrally controlled, the PCECC solution provides a smooth transition way for users.

## 1.2. Using the PCE as the Central Controller (PCECC) Approach

PCECC not only can remove the existing MPLS signaling totally from the control plane without losing any MPLS functionalities, but also will achieve this goal through utilizing the existing PCEP without introducing a new protocol into the network.

The following diagram illustrates the PCECC architecture.



Through the draft, we call the combination of the functionality for global label range signaling and the functionality of LSP setup/download/cleanup using the combination of global labels and local labels as PCECC functionality.

Current MPLS label has local meaning. That is, MPLS label allocated locally and signaled through the LDP/RSVP-TE/BGP etc. dynamic signaling protocol.

As the SDN(Service-Driven Network) technology develops, MPLS global label has been proposed again for new solutions. [I-D.li-mpls-global-label-usecases] proposes possible usecases of MPLS global label. MPLS global label can be used for identification of the location, the service and the network in different application scenarios. From these usecases we can see that no matter SDN or traditional application scenarios, the new solutions based on MPLS global label can facilitate service provisions. The solution choices are described in [I-D.li-mpls-global-label-framework].

To ease the label allocation and signaling mechanism, also with the new applications such as concentrated LSP controller is introduced, PCE can be conveniently used as a central controller and MPLS global label range negotiator.

The later section of this draft describes the user cases for PCE server and PCE clients to have the global label range negotiation and local label range negotiation functionality.

To empower networking with centralized controllable modules, there are many choices for downloading the forwarding entries to the data plane, one way is the use of the OpenFlow protocol, which helps devices to populate their forwarding tables according to a set of instructions to the data plane. There are other candidate protocols to convey specific configuration information towards devices also. Since the PCEP protocol is already deployed in some of the service providers networks, leverage the PCEP to populated the MPLS forwarding table is a possible good choice.

For the centralized network, the performance achieved through distributed system can not be easy matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that the adjacency IDs for all the network nodes and links are propagated through the centralized controller instead of using the IGP extension.

The node and link adjacency IDs can be negotiated through the PCECC with each PCECC clients and these IDs can be just taken from the global label range which has been negotiated already.

With the capability of supporting SR within the PCECC architecture, all the p2p forwarding path protection use cases described in the draft [I-D.ietf-spring-resiliency-use-cases] will be supported too within the PCECC network. These protection alternatives include end-to-end path protection, local protection without operator management and local protection with operator management.

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP LSP using the local label range for each network nodes.

With the capability of setting up/maintaining the P2MP/MP2MP LSP within the PCECC network, it is easy to provide the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.



## 2. Terminology

The following terminology is used in this document.

IGP: Interior Gateway Protocol. Either of the two routing protocols, Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS).

PCC: Path Computation Client: any client application requesting a path computation to be performed by a Path Computation Element.

PCE: Path Computation Element. An entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

TE: Traffic Engineering.

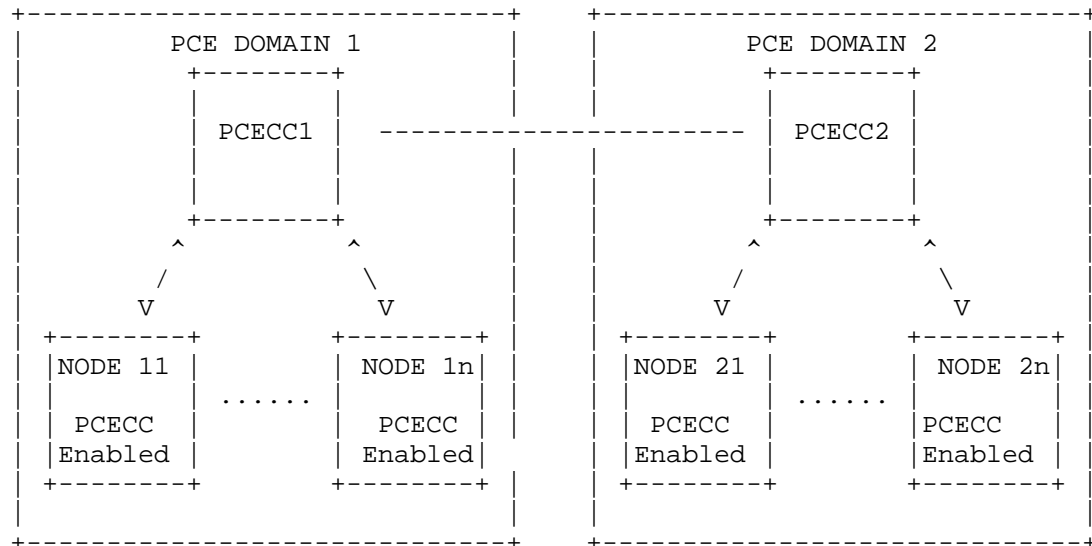
## 3. PCEP Requirements

Following key requirements associated PCECC should be considered when designing the PCECC based solution:

1. Path Computation Element (PCE) clients supporting this draft MUST have the capability to advertise its PCECC capability to the PCECC.
2. Path Computation Element (PCE) supporting this draft MUST have the capability to negotiate a global label range for a group of clients.
3. Path Computation Client (PCC) MUST be able ask for global label range assigned in path request message .
4. PCE are not required to support label reserve service. Therefore, it MUST be possible for a PCE to reject a Path Computation Request message with a reason code that indicates no support for label reserve service.
5. PCEP SHOULD provide a means to return global label range and LSP label assignments of the computed path in the reply message.
6. PCEP SHOULD provide a means to download the MPLS forwarding entry to the PCECC's clients.

#### 4. Use Cases of PCECC for Label Resource Reservations

Example 1 to 2 are based on network configurations illustrated using the following figure:



##### Example 1: Shared Global Label Range Reservation

- o PCECC Clients nodes report MPLS label capability to the central controller PCECC.
- o The central controller PCECC collects MPLS label capability of all nodes. Then PCECC can calculate the shared MPLS global label range for all the PCECC client nodes.
- o In the case that the shared global label range need to be negotiated across multiple domains, the central controllers of these domains need to be communicate to negotiate a common global label range.
- o The central controller PCECC notifies the shared global label range to all PCECC client nodes.

##### Example 2: Global Label Allocation

- o PCECC Client node1 send global label allocation request to the central controller PCECC1.

- o The central controller PCECC1 allocates the global label for FEC1 from the shared global label range and sends the reply to the client node1.
- o The central controller PCECC1 notifies the allocated label for FEC1 to all PCECC client nodes within domain 1.

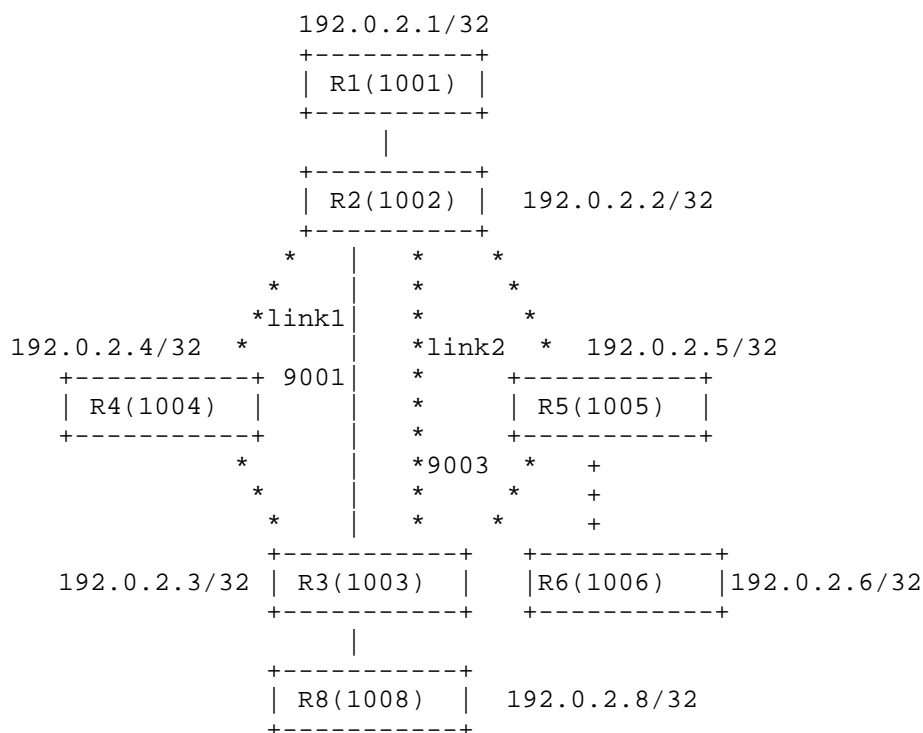
## 5. Using PCECC for SR without the IGP Extension

For the centralized network, the performance achieved through distributed system can not be easily matched if all of the forwarding path is computed, downloaded and maintained by the centralized controller. The performance can be improved by supporting part of the forwarding path in the PCECC network through the segment routing mechanism except that node segment IDs and adjacency segment IDs for all the network are allocated dynamically and propagated through the centralized controller instead of using the IGP extension.

When the PCECC is used for the distribution of the node segment ID and adjacency segment ID, the node segment ID is allocated from the global label pool. For the allocation of adjacency segment ID, there are two choices, the first choice is that it is allocated from the local label pool, the second choice is that it is allocated from the global label pool. The advantage for the second choice is that the depth of the label stack for the forwarding path encoding will be reduced since adjacency segment ID can signal the forwarding path without adding the node segment ID in front of it. In this version of the draft, we use the first choice for now. We may update the draft to reflect the use of the second choice.

Same as the SR solutions, when PCECC is used as the central controller, the support of FRR on any topology can be pre-computed and setup without any additional signaling (other than the regular IGP/BGP protocols) including the support of shared risk constraints, support of node and link protection and support of microloop avoidance.

The following example illustrates the use case where the node segment ID and adjacency segment ID are allocated from the global label allocated for SR path.



### 5.1. Use Cases of PCECC for SR Best Effort(BE) Path

In this mode of the solution, the PCECC just need to allocate the node segment ID and adjacency ID without calculating the explicit path for the SR path. The ingress of the forwarding path just need to encapsulate the destination node segment ID on top of the packet. All the intermediate nodes will forward the packet based on the final destination node segment id. It is similar to the LDP LSP forwarding except that label swapping is using the same global label both for the in segment and out segment in each hop.

The p2p SR BE path examples are explained as bellow:

Note that the node segment id for each node from the shared global labels ranges negotiated already.

Example 1:

R1 may send a packet to R8 simply by pushing an SR header with segment list {1008}. The path can be: R1-R2-R3-R8 or R1-R2-R5-R8 depending on the route calculation on node R2.

#### Example 2: local link/node protection:

For the packet which has destination of R3 and after that, R2 may preinstalled the backup forwarding entry to protect the R4 node, the pre-installed the backup path can go through either node5 or link1 or link2 between R2 and R3. The backup path calculation is locally decided by R2 and any existing IP FRR algorithms can be used here.

### 5.2. Use Cases of PCECC for SR Traffic Engineering (TE) Path

In the case of traffic engineering path is needed, the PCECC need to allocate the node segment ID and adjacency ID, and at the same time PCECC calculates the explicit path for the SR path and pass this explicit path represented with a sequence of node segment id and adjacency id. The ingress of the forwarding path need to encapsulate the stack of node segment id and adjacency id on top of the packet. For the case where strict traffic engineering path is needed, all the intermediate nodes and links will be specified through the stack of labels so that the packet is forwarded exactly as it is wanted.

Even though it is similar to TE LSP forwarding where forwarding path is engineered, but the Qos is only guaranteed through the enforce of the bandwidth admission control. As for the RSVP-TE LSP case, Qos is guaranteed through the link bandwidth reservation in each hop of the forwarding path.

The p2p SR traffic engineering path examples are explained as bellow:

Note that the node segment id for each node is allocated from the shared global labels ranges negotiated already and adjacency segment ids for each link are allocated from the local label pool for each node.

#### Example 1:

R1 may send a packet P1 to R8 simply by pushing an SR header with segment list {1008}. The path should be: R1-R2-R3-R8.

#### Example 2:

R1 may send a packet P2 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.

#### Example 3:

R1 may send a packet P3 to R8 while avoiding the links between R2 and R3 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8

The p2p local protection examples for SR TE path are explained as below:

Example 4: local link protection:

- o R1 may send a packet P4 to R8 by pushing an SR header with segment list {1002, 9001, 1008}. The path should be: R1-R2-(1)link-R3-R8.
- o When node R2 receives the packet from R1 which has the header of R2- (1)link-R3-R8, and also find out there is a link failure of link1, then it will send out the packet with header of R3-R8 through link2.

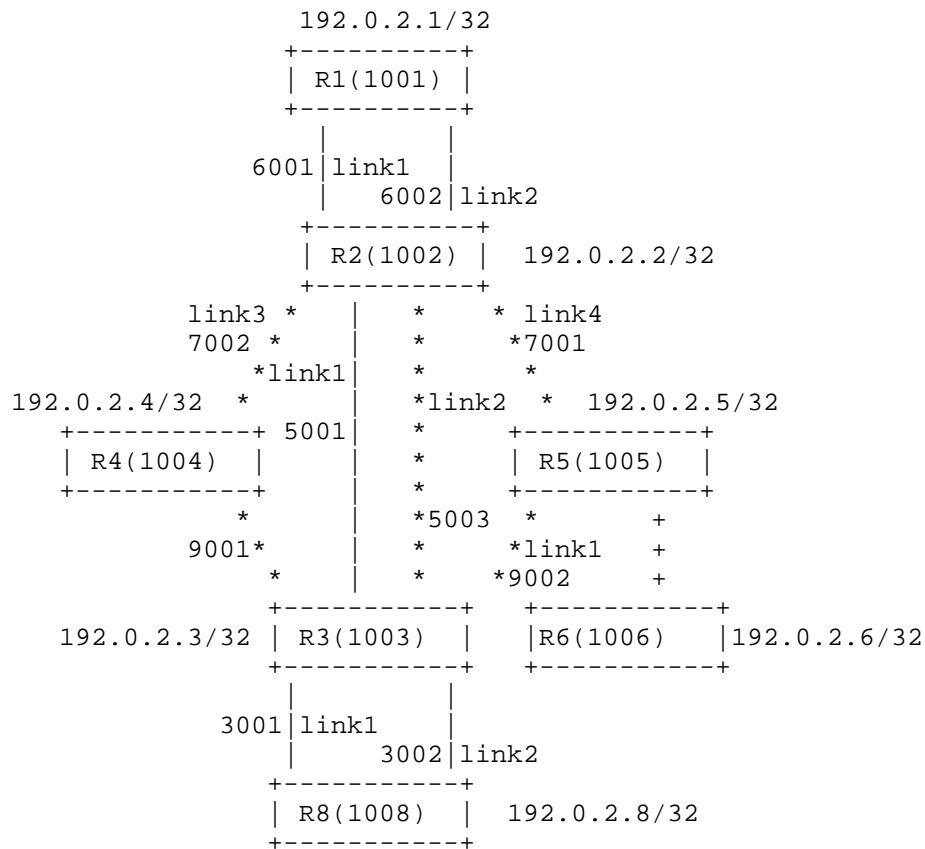
Example 5: local node protection:

- o R1 may send a packet P5 to R8 by pushing an SR header with segment list {1004, 1008}. The path should be : R1-R2-R4-R3-R8.
- o When node R2 receives the packet from R1 which has the header of {1004, 1008}, and also find out there is a node failure for node4, then it will send out the packet with header of {1005, 1008} to node5 instead of node4.

## 6. Use Cases of PCECC for TE LSP

In the previous sections, we have discussed the cases where the SR path is setup through the PCECC. Although those cases give the simplicity and scalability, but there are existing functionalities for the traffic engineering path such as the bandwidth guarantee through the full forwarding path and the multicast forwarding path which SR based solution cannot solve. Also there are cases where the depth of the label stack may have been an issue for existing deployment and certain vendors.

So to address these issues, PCECC architecture should also support the TE LSP and multicast LSP functionalities. To achieve this, the existing PCEP can be used to communicate between the PCE server and PCE's client PCC for exchanging the path request and reply information regarding to the TE LSP info. In this case, the TE LSP info is not only the path info itself, but it includes the full forwarding info. Instead of letting the ingress of LSP to initiate the LSP setup through the RSVP-TE signaling protocol, with minor extensions, we can use the PCEP to download the complete TE LSP forwarding entries for each node in the network.



#### TE LSP Setup Example

- o Node1 sends a path request message for the setup of TE LSP from R1 to R8.
- o PCECC program each node along the path from R1 to R8 with the primary path: {R1, link1, 6001}, {R2, link3, 7002}, {R4, link0, 9001}, {R3, link1, 3001}, {R8}.
- o For the end to end protection, PCECC program each node along the path from R1 to R8 with the secondary path: {R1, link2, 6002}, {R2, link4, 7001}, {R5, link1, 9002}, {R3, link2, 3002}, {R8}.
- o It is also possible to have a secondary backup path for the local node protection setup by PCECC. For example, the primary path is still same as what we have setup so far, then to protect the node R4 locally, PCECC can program the secondary path like this: {R1,

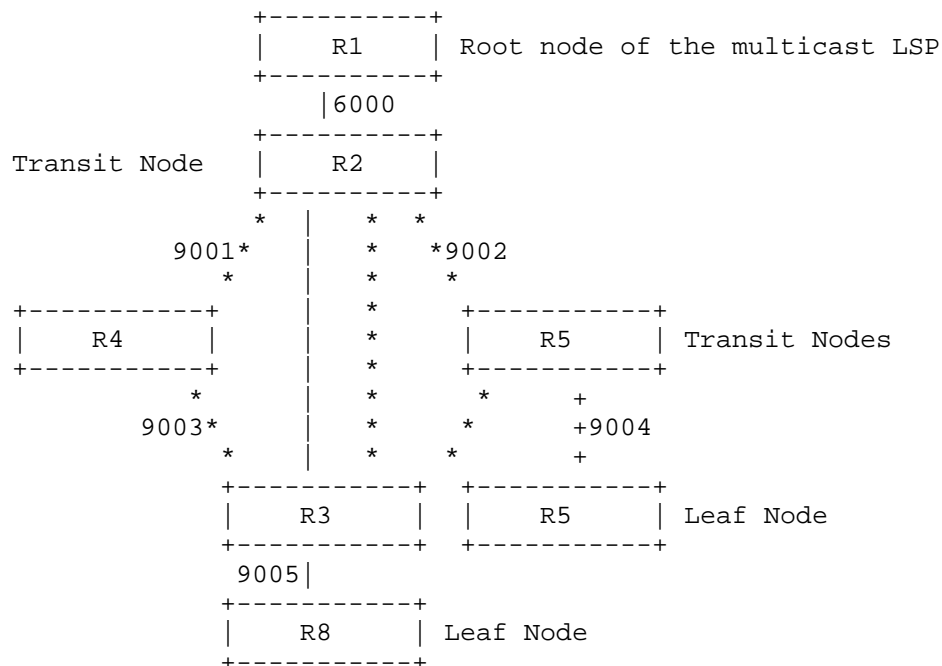
link1, 6001}, {R2, link1, 5001}, {R3, link1, 3001}, {R8}. By doing this, the node R4 is locally protected.

## 7. Use Cases of PCECC for Multicast LSPs

The current multicast LSPs are setup either using the RSVP-TE P2MP or mLDP protocols. The setup of these LSPs not only need a lot of manual configurations, but also it is also complex when the protection is considered. By using the PCECC solution, the multicast LSP can be computed and setup through centralized controller which has the full picture of the topology and bandwidth usage for each link. It not only reduces the complex configurations comparing the distributed RSVP-TE P2MP or mLDP signal lings, but also it can compute the disjoint primary path and secondary path efficiently.

### 7.1. Using PCECC for P2MP/MP2MP LSPs' Setup

With the capability of global label and local label existing at the same time in the PCECC network, PCECC will use compute, setup and maintain the P2MP and MP2MP lsp using the local label range for each network nodes.





The P2MP examples are explained here:

Step1: R1 may send a packet P1 to R2 simply by pushing an label of 6000 to the packet.

Step2: After R2 receives the packet with label 6000, it will forwarding to R4 by pushing header of 9001 and R5 by pushing header of 9002.

Step3: After R4 receives the packet with label 9001, it will forwarding to R3 by pushing header of 9003. After R5 receives the packet with label 9002, it will forwarding to R5 by pushing header of 9004.

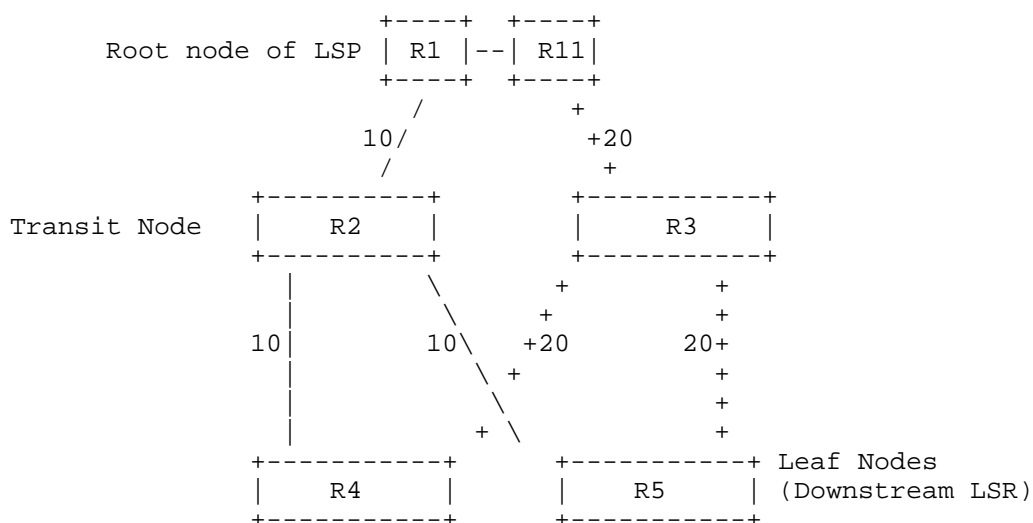
Step3: After R3 receives the packet with label 9003, it will forwarding to R8 by pushing header of 9005

## 7.2. Use Cases of PCECC for the Resiliency of P2MP/MP2MP LSPs

### 7.2.1. PCECC for the End-to-End Protection of the P2MP/MP2MP LSPs

In this section we describe the end-end managed path protection service and the local protection with the operation management in the PCECC network for the P2MP/MP2MP LSP, which includes both the RSVP-TE P2MP based LSP and also the mLDP based LSP.

An end-to-end protection (for nodes and links) principle can be applied for computing backup P2MP or MP2MP LSPs. During computation of the primarily multicast trees, PCECC server may also be taken into consideration to compute a secondary tree. A PCE may compute the primary and backup P2MP or MP2MP LSP together or sequentially.

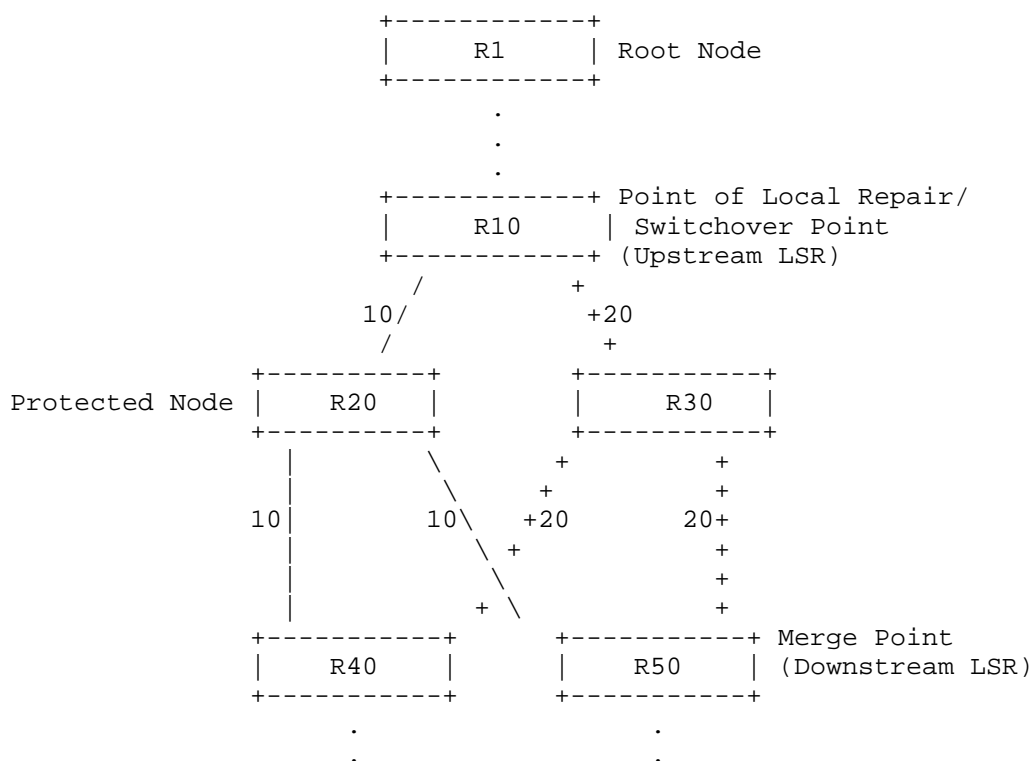


In the example above, when the PCECC setup the primary multicast tree from the root node R1 to the leafs, which is R1->R2->{R4, R5}, at same time, it can setup the backup tree, which is R1->R11->R3->{R4, R5}. Both the these two primary forwarding tree and secondary forwarding tree will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R1->R2->{R4, R5} path normally, and when there is a node in the primary tree, then the root node R1 will switch the flow to the backup tree, which is R1->R11->R3->{R4, R5}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

#### 7.2.2. PCECC for the Local Protection of the P2MP/MP2MP LSPs

In this section we describe the local protection service in the PCECC network for the P2MP/MP2MP LSP.

While the PCECC sets up the primary multicast tree, it can also build the back LSP among PLR, the protected node, and MPs (the downstream nodes of the protected node). In the cases where the amount of downstream nodes are huge, this mechanism can avoid unnecessary packet duplication on PLR, so that protect the network from traffic congestion risk.



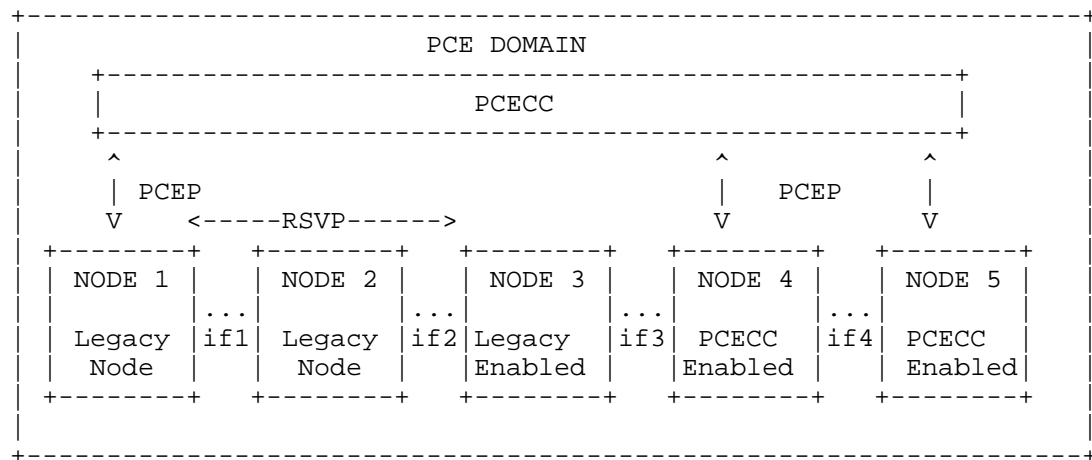
In the example above, when the PCECC setup the primary multicast path around the PLR node R10 to protect node R20, which is R10->R20->{R40, R50}, at same time, it can setup the backup path R10->R30->{R40, R50}. Both the these two primary forwarding path and secondary forwarding path will be downloaded to each routers along the primary path and the secondary path. The traffic will be forwarded through the R10->R20->{R40, R50} path normally, and when there is a node failure for node R20, then the PLR node R10 will switch the flow to the backup path, which is R10->R30->{R40, R50}. By using the PCECC, the path computation and forwarding path downloading can all be done without the complex signaling used in the P2MP RSVP-TE or mLDP.

#### 8. Use Cases of PCECC for LSP in the Network Migration

One of the main advantages for PCECC solution is that it has backward compatibility naturally since the PCE server itself can function as a proxy node of MPLS network for all the new nodes which don't support the existing MPLS signaling protocol anymore.

As it is illustrated in the following example, the current network will migrate to a total PCECC controlled network gradually by replacing the legacy nodes. During the migration, the legacy nodes still need to signal using the existing MPLS protocol such as LDP and RSVP-TE, and the new nodes setup their portion of the forwarding path through PCECC directly. With the PCECC function as the proxy of these new nodes, MPLS signaling can populate through network as normal.

Example described in this section is based on network configurations illustrated using the following figure:



#### Example: PCECC Initiated LSP Setup In the Network Migration

In this example, there are five nodes for the TE LSP from head end (Node1) to the tail end (Node5). Where the Node4 and Node5 are centrally controlled and other nodes are legacy nodes.

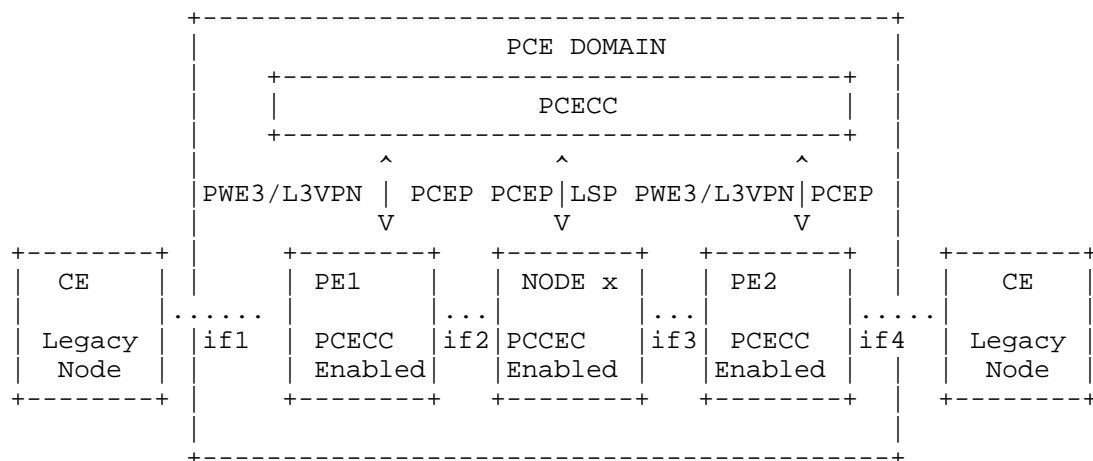
- o Node1 sends a path request message towards PCECC for the setup of LSP destinating to Node5.
- o PCECC sends to nodel a reply message for LSP setup with the path: (Node1, if1), (Node2, if2), (Node3, if3), (Node4, if4), Node5.
- o Node1, Node2, Node3 will setup the LSP to Node5 using the local labels as usual.
- o Then the PCECC will program the outsegment of Node3, the insegment/ ousegment of Node4, and the insegment for Node5.

## 9. Use Cases of PCECC for L3VPN and PWE3

The existing services using MPLS LSP tunnels based on MPLS signalling mechanism such L3VPN, PWE3 and IPv6 can be simplified by using the PCECC to negotiate the label assignments for the L3VPN, PWE3 and Ipv6.

In the case of L3VPN, VPN labels can be negotiated and distributed through the PCECC PCEP among the PE router instead of using the BGP protocols.

Example described in this section is based on network configurations illustrated using the following figure:



Example: Using PCECC for L3VPN and PWE3

In the case of PWE3, instead of using the LDP signalling protocols, the label and port pairs assigned to each pseudowire can be negotiated through PCECC among the PE routers and the corresponding forwarding entries will be distributed into each PE router through the extended PCEP protocols.

## 10. Using PCECC for Traffic Classification Information

When a TE-LSP is set up, the head end needs to know:

- o how to use it

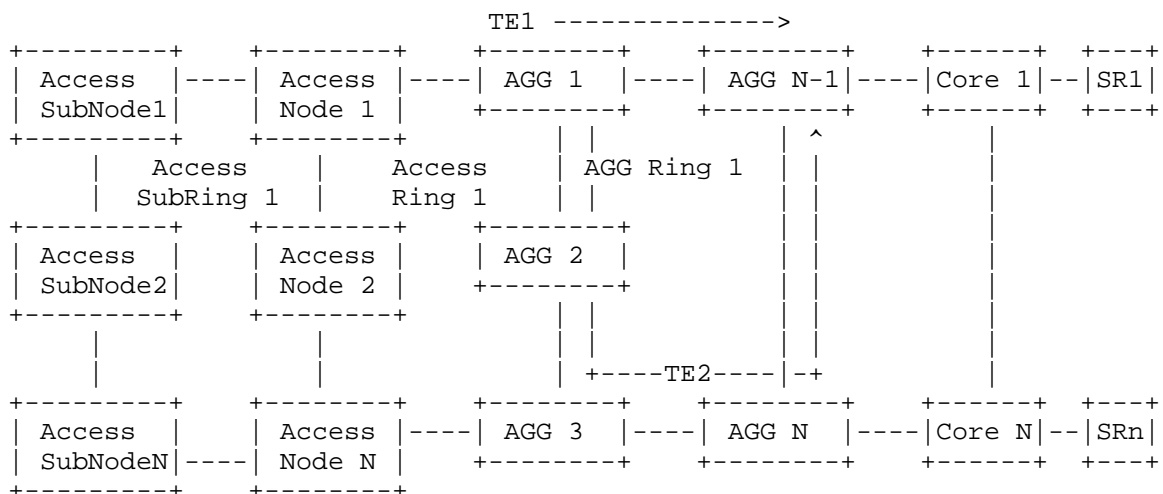
- o What traffic to send on the LSP
- o Whether it is a virtual link
- o Whether to advertise it in the IGP
- o What bits of this information to signal to the tail end

PCEP allows an Active PCE to set up or modify LSPs. But we have no way to tell the head end how to use the LSP. This is because of history. It used to be the LER that made the request of the PCE, so it knew why it wanted the LSP.

With the PCECC architecture by extending the PCEP protocols, it is easy to carry this information such as how to use the LSP, how to advertise the LSP and other extra signaling information.

#### 11. PCECC Load Balancing (LB) Use Case

Very often many service providers use TE tunnels for solving issues with non-deterministic paths in their networks. One example of such applications is usage of TEs in the mobile backhaul (MBH). Let's consider the following typical topology.



This MBH architecture uses L2 access rings and subrings. L3 starts at aggregation. For the sake of simplicity here we have only one access subring, access ring and aggregation ring (AGG1...AGGN), connected by Nx10GE interfaces. Aggregation domain runs its own IGP. There are two Egress routers (AGG N-1, AGG N) that are connected to the Core domain via L2 interfaces. Core also have connections to service routers,

RSVP TEs are used for MPLS transport inside the ring. There could be at least 2 tunnels (one way) from each AGG router to egress AGG routers. There are also many L2 access rings connected to AGG routers.

Service deployment made by means of either L2VPNs (VPLS) or L3VPNs. Those services use MPLS TE as transport towards egress AGG routers. TE tunnels could be also used as transport towards service routers in case of seamless MPLS based architecture in the future.

There is a need to solve the following tasks:

- o Perform automatic LB amongst TE tunnels according to current traffic load
- o TE bandwidth (BW) management: Provide guaranteed BW for specific service: HSI, IPTV, etc., provide time-based BW reservation (BoD)
- o Simplify development of TE tunnels (go away from manual provisioning)
- o Provide flexibility for Service Router placement (anywhere in the network by creation of transport LSPs to them)

Since other tasks are considered in other PCECC use cases above, hereafter we will focus only on load balancing (LB) task. LB task could be solved by means of PCECC in the following way:

- o After application or network service or operator will ask SDN controller (PCECC) for LSP based LB between AGG X and AGG N/AGG N-1 (egress AGG routers which have connections to core) via North Bound Interface (NBI such as REST API), PCECC SHOULD ask for constrains for that particular calculation (i.e. LSP type: traditional CR-LSP or SR-TE LSP, bandwidth, inclusion or exclusion specific links or nodes, number of paths, shortest path or minimum cost tree, need for disjoint LSP paths etc.).
- o PCECC MUST calculate N P2P LSPs according to given constrains, calculation is based on results of Objective Function (OF), that includes same source and destination routers IDs, same or different bandwidth (BW), different links (in case of disjoint paths) and other constrains from Step 1.
- o Depending on given LSP type (CR-LSP or SR-TE), PCECC SHOULD create different labels (aka different label spaces, it MAY also require label space negotiation procedure between PCECC and PCCs) for calculated LSPs from egress nodes AGG N-1 and AGG N towards ingress AGG X node.
- o PCECC SHOULD send PCInitiate PCEP message [I-D.crabbe-pce-pce-initiated-lsp] towards ingress AGG X router(PCC) for each of N LSPs and receives PCRpt PCEP message [I-D.ietf-pce-stateful-pce] back from him.
- o If LSP type is CR-LSP, PCECC MUST send PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce-controller] PCEP message to each node along the path with label information for each of N LSPs. If LSP type is SR-TE, PCECC also MUST send PCLabelUpd PCEP message

to each node along the path with label information (Node-ID and Adjacency-ID segment (label) list) specific to that node. Then PCECC SHOULD send PCUpd PCEP message to the ingress AGG X router with information about new LSP and AGG X(PCC) SHOULD send PCEP PCrpt back with LSP status:Up.

- o Now each router along the LSP has corresponding label forwarding state for each of N LSPs.

- o AGG X as ingress router now have N LSPs towards AGG N and AGG N-1 which are available for installing to router's RIB and LB of traffic between them. Traffic distribution between those LSPs depends on particular realization of hash-function on that router.

- o Since PCECC MUST know as LSDB as TEDB (TE state) he can manage and prevent possible oversubscriptions and limit number of available LB states.

## 12. Using reliable P2MP TE based multicast delivery for distributed computations (MapReduce-Hadoop)

MapReduce model of distributed computations in computing clusters is widely deployed. In Hadoop 1.0 architecture MapReduce operations on big data performs by means of Master-Slave architecture in the Hadoop Distributed File System (HDFS), where NameNode has the knowledge about resources of the cluster and where actual data (chunks) for particular task are located (which DataNode). Each chunk of data (64MB or more) should have 3 saved copies in different DataNodes based on their proximity.

Proximity level currently has semi-manual allocation and based on Rack IDs (Assumption is that closer data are better because of access speed/smaller latency).

JobTracker node is responsible for computation tasks, scheduling across DataNodes and also have Rack-awareness. Currently transport protocols between NameNode/JobTracker and DataNodes are based on IP unicast. It has simplicity as pros but has numerous drawbacks related with its flat approach.

It is clear that we should go beyond of one DC for Hadoop cluster creation and move towards distributed clusters. In that case we need to handle performance and latency issues.

Latency depends on speed of light in fiber links and also latency introduced by intermediate devices in between. The last one is closely correlated with network device architecture and performance. Current performance of NPU based routers should be enough for creating distribute Hadoop clusters with predicted latency. Performance of SW based routers (mainly as VNF) together with additional HW features such as DPDK are promising but require additional research and testing.



Main question is how can we create simple but effective architecture for distributed Hadoop cluster?

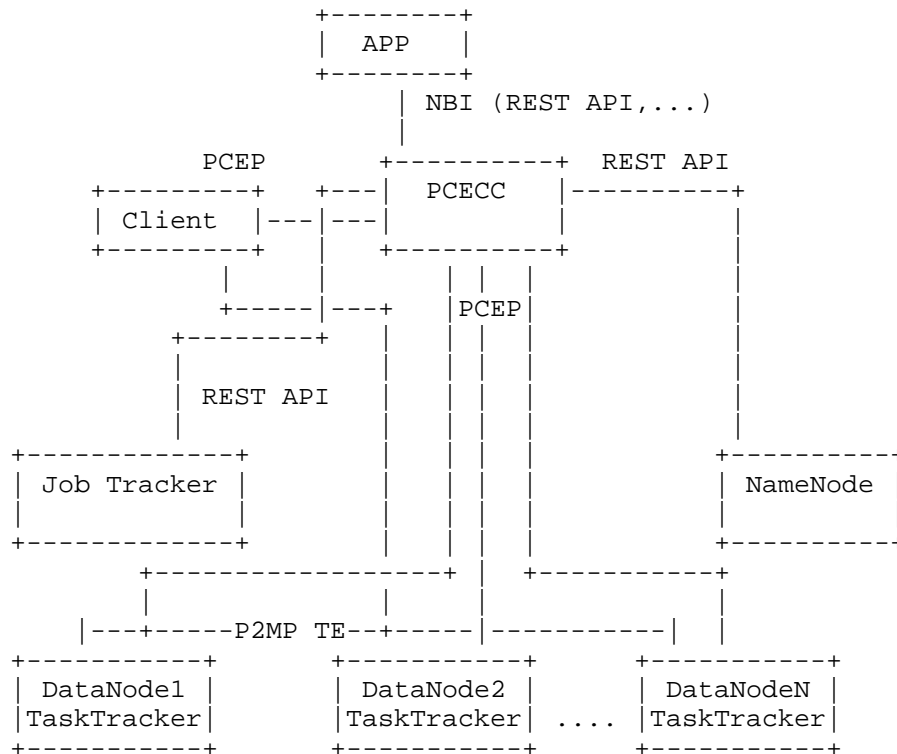
There are number of researches [Multicast Tree Map-Reduce...] which show how usage of multicast tree could improve speed of resource or cluster members discovery inside the cluster as well as increase redundancy in communications between cluster nodes.

Is traditional IP based multicast enough for that? We doubt it because it requires additional control plane (IGMP, PIM) and a lot of signaling, that is not suitable for high performance computations, that are very sensitive to latency.

P2MP TE tunnels looks much more suitable as potential solution for creation of multicast based communications between Master and Slave nodes inside cluster. Obviously these P2MP tunnels should be dynamically created and turned down (no manual intervention). Here is there PCECC comes to play. His main task is to create optimal topology of each particular request for MapReduce computation and also create P2MP tunnels with needed parameters such as badnwidth and delay.

This solution would require to use MPLS label based forwarding inside the cluster. Usage of label based forwarding inside DC was proposed by Yandex [MPLS in DC...] Technically it is already possible because mpls on switches is already supported by some vendors, mpls aslo exists on Linux and OVS.

The following framework can make this task:



Communication between Master nodes (JobTracker and NameNode) and PCECC via REST API MAY be either done directly or via cluster manager such as Mesos.

### Phase 1: Distributed cluster resources discovery

During this phase Master Nodes SHOULD identify and find available Slave nodes according to computing request from application (APP). NameNode SHOULD query PCECC about available DataNodes, NameNode MAY provide additional constraints to PCECC such as topological proximity, redundancy level.

PCECC SHOULD analyze the topology of distributed cluster and perform constrain based path calculation [RFC7334] from client towards most suitable NameNodes. PCECC SHOULD reply to NameNode the list of most suitable DataNodes and their resource capabilities. Topology discovery mechanism for PCECC will be added later to that framework.

Phase 2: PCECC SHOULD create P2MP LSP from client towards those DataNodes by means of PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce-controller] PCEP messages following previously calculated path.

Phase 3. NameNode SHOULD send this information to client, PCECC informs client about optimal P2MP path towards DataNodes via PCEP PCUpd message.

Phase 4. Client sends data blocks to those DataNodes for writing via created P2MP tunnel.

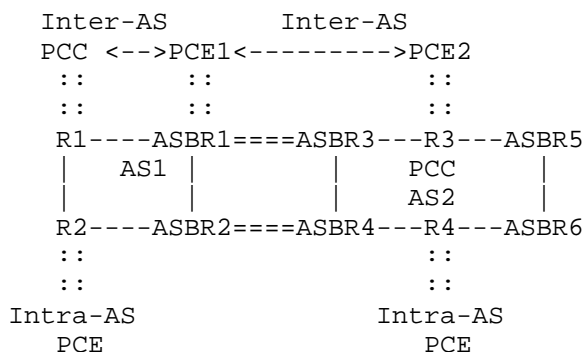
When this task will be finished, P2MP tunnel MAY be turned down.

### 13. PCECC and Inter-AS TE

There are three signalling options for establishing Inter-AS TE LSP: contiguous TE LSP [RFC5151], stitched inter-AS TE LSP [RFC5150], nested TE LSP [RFC4206].

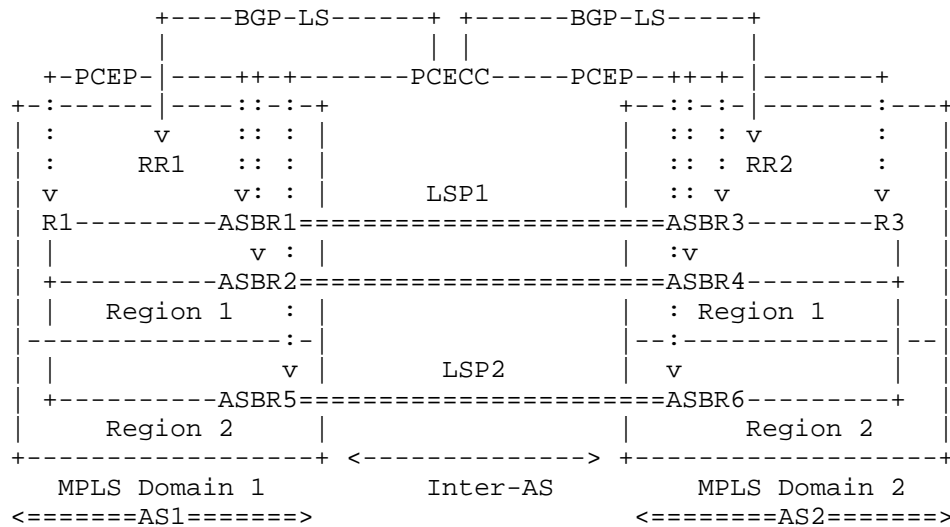
Requirements for PCE-based Inter-AS setup [RFC5376] describe the approach and PCEP functionality that are needed for establishing Inter-AS TE LSPs.

[RFC5376] also gives Inter- and Intra-AS PCE Reference Model that is provided below in shorten form for the sake of simplicity.



Shorten form of Inter- and Intra-AS PCE Reference Model [RFC5376]

Hereafter we will discuss a simplified Inter-AS case when both AS1 and AS2 belong to the same service provider administration. In that case Inter and Intra-AS PCEs could be combined in one single PCE if such combined PCE performance is enough for handling all Path Computation Requests. Even more in that particular case we potentially could use single PCE for both ASes if his scalability and performance are enough, we just will need interfaces (PCEP and BGP-LS) to both domains. SDN controller's redundancy mechanisms are out of scope in our case. Thus routers in AS1 and AS2 (PCCs) will send Path Computation Requests towards same PCE.



Particular case of Inter-AS PCE Reference Model

In one particular case of PCECC Inter-AS TE scenario service provider controls both domains (AS1 and AS2), each of them have own IGP and MPLS transport. The need is to setup Inter-AS LSPs for transporting different services on top of them (Voice, L3 VPN etc.) Inter-AS links with different capacity exist in several regions. The task is not only to provision those Inter-AS LSPs with given constraints but also calculate the path and pre-setup the backup Inter-AS LSPs that will be used if main LSP fails.

For the figure above it would be that LSP1 from R1 to R3 SHOULD go via ASBR1 and ASBR3, and it is the main Inter-AS LSP. R1-R3 LSP2 that SHOULD go via ASBR5 and ASBR6 is the backup one. Depending on Inter-AS TE type, backup LSP could be used either by head-end R1 or ASBR1.

After the addition of PCECC functionality to PCE (SDN controller), PCECC based Inter-AS TE model SHOULD follow as PCECC usecase for TE LSP (case 6 above) as requirements of [RFC5376] with the following details:

- o Since PCECC MUST know the topology of both domains AS1 and AS2, PCECC MUST establish BGP-LS peering with routers (or RRs) in both domains
- o PCECC MUST have SBI (PCEP) connectivity towards all routers in both domains (see also section 4 in [RFC5376])
- o After operator's application or service orchestrator will create request for topology of specific service, PCECC SHOULD receive that request via NBI (NBI type is implementation dependent, MAY be NETCONF/Yang, REST etc.). Then PCECC SHOULD calculate Objective Function (OF) for optimal path with given constraints (i.e. LSP type, bandwidth etc.), including those from [RFC5376]: priority, AS sequence, preferred ASBR, disjoint paths, protection. On this step we would have two paths: R1-ASBR1-ASBR3-R3, R1-ASBR5-ASBR6-R3
- o Depending on given LSP type (CR-LSP or SR-TE), PCECC SHOULD create different labels (aka different label spaces, it MAY also require label space negotiation procedure between PCECC and PCCs) for calculated LSPs from egress node in one AS towards ingress in another AS.
- o PCECC SHOULD send PCInitiate PCEP message [I-D.crabbe-pce-pce-initiated-lsp] towards ingress router R1 (PCC) in AS1 and receive PCRpt PCEP message [I-D.ietf-pce-stateful-pce] back from him.
- o If LSP type is CR-LSP, PCECC MUST send PCLabelUpd [I-D.zhao-pce-pcep-extension-for-pce-controller] PCEP message to each node along the path (ASBR1-ASBR3-R3, ASBR5-ASBR6-R3) in both ASes with label information for that LSP.
- o If LSP type is SR-TE, PCECC also MUST send PCLabelUpd PCEP message to each node along the path in both ASes with label information (Node-ID and Adjacency-ID segment (label) list) specific to that node.
- o Then PCECC SHOULD send PCUpd PCEP message to the ingress router R1 in AS1 with information about new LSP and the R1 router SHOULD send PCEP PCRpt back

with LSP1 and LSP2 status:Up.

- o After that step R1 SHOULD have main and backup TEs (LSP1 and LSP2) towards R3 up. It is up to implementation how to put this TEs to R1's RIB and how to make switchover to backup LSP2 if LSP1 fails.

#### 14. The Considerations for PCECC Procedure and PCEP extensions

The PCECC's procedures and PCEP extensions is defined in [I-D.zhao-pce-pcep-extension-for-pce-controller].

#### 15. IANA Considerations

This document does not require any action from IANA.

Zhao, et al.

Expires May 25, 2017

[Page 25]

## 16. Security Considerations

TBD.

## 17. Acknowledgments

We would like to thank Robert Tao, Changjiang Yan, Tieying Huang, Adrian Farrel, Sergio Belotti and Dieter Beller, Andrey Elperin and Evgeniy Brodskiy for their useful comments and suggestions.

## 18. References

### 18.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<http://www.rfc-editor.org/info/rfc5440>>.

### 18.2. Informative References

- [RFC5441] Vasseur, JP., Ed., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, DOI 10.17487/RFC5441, April 2009, <<http://www.rfc-editor.org/info/rfc5441>>.
- [RFC5541] Le Roux, JL., Vasseur, JP., and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, DOI 10.17487/RFC5541, June 2009, <<http://www.rfc-editor.org/info/rfc5541>>.
- [RFC5376] N. Bitar, R. Zhang, K. Kumaki "Inter-AS Requirements for the Path Computation Element Communication Protocol (PCECP)", RFC 5376, DOI 10.17487/RFC5376, November 2008 <<http://www.rfc-editor.org/info/rfc5376>>.
- [I-D.filsfils-spring-segment-routing] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.
- [I-D.ietf-pce-stateful-pce] Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-14 (work in progress), May 2016.
- [I-D.crabbe-pce-pce-initiated-lsp] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-crabbe-pce-pce-initiated-lsp-05 (work in progress), October 2015.
- [I-D.ali-pce-remote-initiated-gmpls-lsp]

Ali, Z., Sivabalan, S., Filsfils, C., Varga, R., Lopez, V., Dios, O., and X. Zhang, "Path Computation Element Communication Protocol (PCEP) Extensions for remote-initiated GMPLS LSP Setup", draft-ali-pce-remote-initiated-gmpls-lsp-03 (work in progress), February 2014.

[I-D.ietf-isis-segment-routing-extensions]

Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-06 (work in progress), December 2015.

[I-D.psenak-ospf-segment-routing-extensions]

Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-psenak-ospf-segment-routing-extensions-05 (work in progress), June 2014.

[I-D.sivabalan-pce-segment-routing]

Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., Raszuk, R., Lopez, V., and J. Tantsura, "PCEP Extensions for Segment Routing", draft-sivabalan-pce-segment-routing-03 (work in progress), July 2014.

[I-D.li-mpls-global-label-usecases]

Li, Z., Zhao, Q., Yang, T., Raszuk, R., and L. Fang, "Usecases of MPLS Global Label", draft-li-mpls-global-label-usecases-03 (work in progress), October 2015.

[I-D.li-mpls-global-label-framework]

Li, Z., Zhao, Q., Chen, X., Yang, T., and R. Raszuk, "A Framework of MPLS Global Label", draft-li-mpls-global-label-framework-02 (work in progress), July 2014.

[I-D.zhao-pce-pcep-extension-for-pce-controller]

Zhao, Q., Li, Z., Dhody, D., and C. Zhou, "PCEP Procedures and Protocol Extensions for Using PCE as a Central Controller (PCECC) of LSPs", draft-zhao-pce-pcep-extension-for-pce-controller-03 (work in progress), March 2016.

[I-D.ietf-spring-resiliency-use-cases]

Francois, P., Filsfils, C., Decraene, B., and R. Shakir, "Use-cases for Resiliency in SPRING", draft-ietf-spring-resiliency-use-cases-02 (work in progress), December 2015.

[MPLS in DC...]

Afanasiev, D., Ginsburg, D., "MPLS in DC and inter-DC networks: the unified forwarding mechanism for network programmability at scale "

[Multicast Tree Map-Reduce...]

Lee, Kyungyong., Dr. Boykin, P. Oscar., Dr.Figueiredo, Renato J., "Multicast Tree Map-Reduce: Self-organizing Resource Discovery and Monitoring using Structured P2P Systems"

Authors' Addresses

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

EMail: quintin.zhao@huawei.com



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

EMail: lizhenbin@huawei.com

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

EMail: khasanov.boris@huawei.com

King Ke  
Tencent Holdings Ltd.  
Shenzhen  
China

EMail: kinghe@tencent.com

Luyuan Fang  
Microsoft

EMail: lufang@microsoft.com

Chao Zhou  
Cisco Systems

EMail: chao.zhou@cisco.com

Boris Zhang  
Telus Communications

EMail: Boris.zhang@telus.com

Artem Rachitskiy  
Mobile TeleSystems JLLC  
Nezavisimosti ave., 95  
Minsk 220043  
Belarus

EMail: arachitskiy@mts.by

Anton Gulida  
Mobile TeleSystems JLLC  
Nezavisimosti ave., 95  
Minsk 220043  
Belarus

EMail: agulida@mts.by



TEAS Working Group  
Internet Draft  
Intended status: Informational

Haomian Zheng  
Young Lee  
Huawei  
Daniele Ceccarelli  
Ericsson  
Jong Yoon Shin  
SK Telecom  
Yunbin Xu  
CAICT  
Yunbo Li  
China Mobile  
Yan Shi  
China Unicom  
Boyuan Yan  
BUPT

Expires: April 30 2017

October 31, 2016

Inter-op Test Cases in Multi-vendor Scenario based on ACTN Architecture

draft-zheng-teas-actn-multivendor-interop-00

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 April 30, 2017.

## Copyright Notice

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

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

## Abstract

ACTN is a practical approach to repurpose existing and well-defined technologies, and underpinning them with SDN principle. It provides a hierarchal architecture to scale and support multi-vendor multi-domain interworking using RESTconf(YANG Model)/PCEP/BGP-LS.

This document contains a test case proposal focused multi-domain, multi-vendor interoperation test cases for the ACTN framework. These test cases cover four test scenarios, including topology abstraction, E2E service provisioning, DC load balancing and inter-domain restoration, to demonstrate the fundamental ideas of the ACTN framework and standards.

## Conventions used in this document

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

## Table of Contents

|                       |   |
|-----------------------|---|
| 1. Introduction ..... | 3 |
|-----------------------|---|

|                                                            |    |
|------------------------------------------------------------|----|
| 2. Terminology .....                                       | 3  |
| 3. Related work: Architecture, Modeling and Protocols..... | 4  |
| 4. Test-cases .....                                        | 4  |
| 4.1. Topology Abstraction .....                            | 4  |
| 4.2. E2E Service Provisioning .....                        | 6  |
| 4.3. DC Load Balancing .....                               | 8  |
| 4.4. Inter-domain Recovery .....                           | 8  |
| 5. Implementation Details.....                             | 8  |
| 6. Future work .....                                       | 9  |
| 7. Security Considerations.....                            | 9  |
| 8. References .....                                        | 9  |
| 8.1. Informative References.....                           | 9  |
| 9. Contributors' Addresses.....                            | 10 |
| 10. Acknowledgment .....                                   | 12 |

## 1. Introduction

The ACTN interoperation test cases are designed to demonstrate the on-going work in IETF of the ACTN framework, including:

- ACTN basic solutions for SDN architecture to support multi-domain, multi-vendor supporting.
- ACTN important interfaces are focused on IETF standards for multi-protocol supporting.

This document is focused on the demonstration of interoperation test case procedures to show how ACTN architecture can be implemented. The ACTN hierarchical controllers include Customer Network Controller (CNC), the Multi-domain Service Coordinator (MDSC), and Physical Network Controller (PNC). In this interoperation test, we focus on the MDSC, PNC, and the MDSC-PNC Interface (MPI) between them. The ACTN MPI can support both RESTCONF/YANG and PCEP-LS and gives a deployment choice that meets the need of the operators.

## 2. Terminology

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

### 3. Related work: Architecture, Modeling and Protocols

ACTN provides description about future requirements in SDN literature in [ACTN-Requirements]. ACTN framework has been proposed in [ACTN-Frame] to support all the requirements. The information model has been defined in [ACTN-Info].

From the solution perspective, ACTN work has been associated with some other IETF works in various working group including netmod, i2rs, rtgwg, ccamp, pce and so on. Yang models, defined in Netconf [RFC6241]/Restconf [Restconf], have been considered as an approach for network configuration. [ACTN-YANG] has described the applicability of YANG models in the ACTN framework, where various yang models including TE topology model from [TE-Topology], tunnel model from [TE-Tunnel] and service model from [Transport-Service-Model] and [ACTN-VN-YANG] have been applied on the ACTN interfaces to complete different functions. PCE protocol in the pce working group has also been considered to be extended and applied for the ACTN interfaces. The applicability draft can be found in [ACTN-PCE]. PCEP extensions with Link State features can be found in [PCEP-LS] for topology discovery, and PCE Initiation mechanism defined in [PCE-Init] can be used to set up the connections.

### 4. Test-cases

This section provides a number of test cases. Currently the test is basically conducted between the MDSC and PNC, i.e., on the MPI interface in the ACTN architecture. The scenario we are going to test includes the topology abstraction, E2E service provisioning, DC load balancing and Inter-domain recovery.

Some fundamental environment has been set up and applied to all the test cases. On the network element level, emulators from various vendors have been connected with their corresponding PNC. The interface between PNC and network elements is known as the southbound interface (SBI) in SDN literature. The protocol on SBI can be vendor-specific.

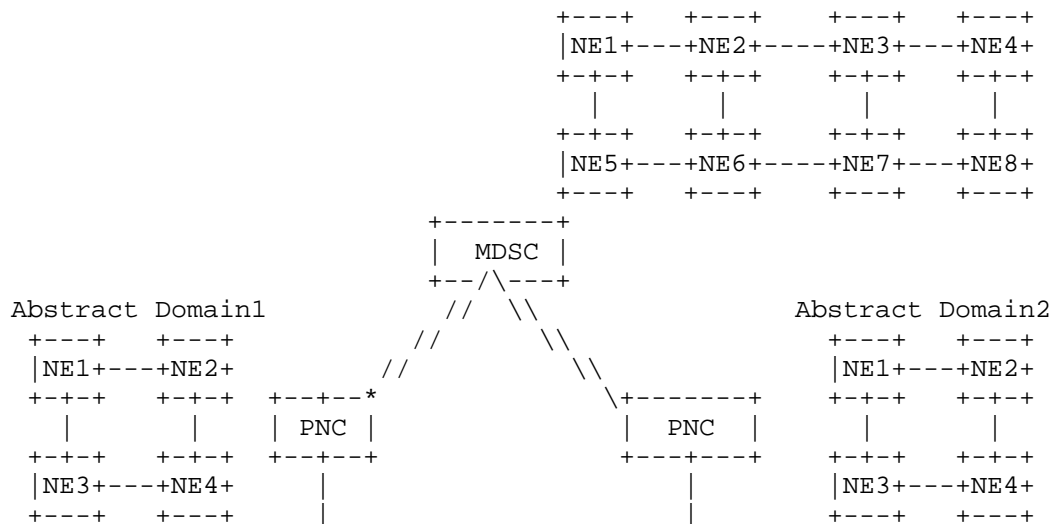
The MDSC can be either from network operators or vendors, to coordinate among multiple PNCs. The interaction between the MDSC and the PNCs, which is known as MPI in ACTN and considered as a part of northbound interface in SDN literature, should be standard.

#### 4.1. Topology Abstraction

This scenario is used to verify the multi-domain topology collection on the MDSC level. Multi-technology network (IP, MPLS, Transport)

should report their topology and the MDSC should be capable of integrating different topologies together.

MDSC, PNC and network elements are involved in this scenario. The PNC needs to collect the information from the network elements and maintain its own TE Database. Given the physical topology, the PNC may simplify the topology and report an 'abstract' version to the MDSC. The interaction on MPI to exchange the topology information may be via RESTconf with topology YANG model. The example of topology abstraction is shown in Fig. 1, PNC collect the information of 6 network elements in every single domain, and abstract the topology into a 4-point format. Then the abstracted topology can be reported to MDSC. In this example, MDSC will receive the information from two PNC domains, and manipulate on an 8-point abstracted topology.



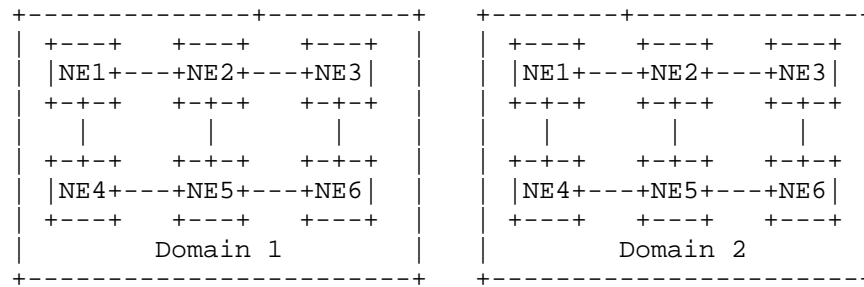
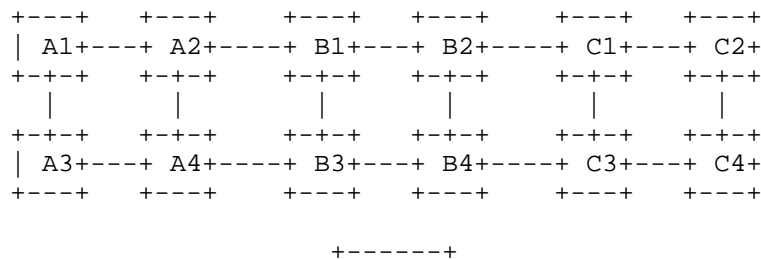


Fig. 1 Topology Abstraction Scenario

Dynamical changes in topology should also be considered. For example when there is a new node discovered in the network, the incremental topology should be correctly detected on PNC, and the abstraction may need to be adjusted accordingly and reported to MDSC for update.

#### 4.2. E2E Service Provisioning

This test case is used to verify the Restconf/YANG functionality on service provisioning via interaction between MDSC and PNC. Inter-domain connection, shown in Fig. 2, is expected to be established. In Fig. 2 the topology on the MDSC is shown, and the network element information is hidden.





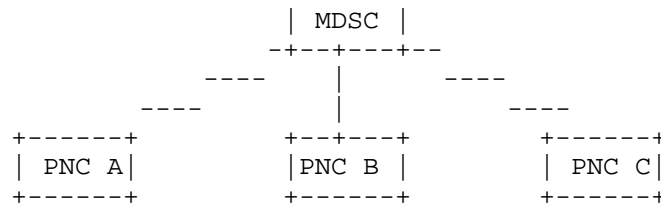


Fig. 2: E2E Service Provisioning Scenario

We assume there is request of 100GE service from A1 to C2, following steps need to be completed to set up the connection.

Step 1: MDSC compute an inter-domain path and translate the multi-domain request into multiple single domain requests of domain A, B and C.

Step 2: MDSC sends decomposed requests to each PNC, and each PNC compute intra-domain path:

- a) Send request to PNC of domain A to set up a service from A1 to A2.
- b) Send request to PNC of domain B to set up a service from B1 to B2.
- c) Send request to PNC of domain C to set up a service from C1 to C2.
- d) These requests may be sent in parallel.

Step 3: Each PNC responds successfully creation, or failure with error message.

Step 4: MDSC receives responds from all the PNCs, and successfully set up a multi-domain service.

The communication between MDSC and PNC could be completed by Restconf protocol associated with tunnel YANG model or service YANG model. The communication between PNC and network elements can be vendor-specific in this scenario.

#### 4.3. DC Load Balancing

DC Load balancing is more advanced scenario based on active LSP set up in the network. In this scenario, the result of previous 2 scenarios needs to be reused. We assumed there are two disjoint LSPs, one from A1 to C2, and another from A1 to C4. Explicit route can be found as following:

LSP1: A1-A2-B1-B2-C1-C2;

LSP2: A1-A3-A4-B3-B4-C3-C4;

The bandwidth of LSP1 and LSP2 are set to 300G and 100G respectively before load balancing, with a target on adjusting both of them to 200G for load balancing. MDSC need to send update message to PNCs to adjust their bandwidth on corresponding links.

The interaction on MPI in this case can be completed by Restconf protocol with topology and service YANG model. The interactions between PNC and network elements are vendor-specific.

#### 4.4. Inter-domain Recovery

Another test case about inter-domain recovery is also designed to verify the function of cross-domain restoration. In this case a cross-domain LSP is set up, such as reusing the LSP1 in section 4.3. Then a failure in one domain is emulated in one domain, and the PNC of this domain cannot find sufficient resources within the domain so that it MUST turn to the MDSC for inter-domain restoration. MDSC then need to update the topology and resource usage in every domain to compute a path for re-routing. After MDSC got an answer, it will deliver another E2E service, which is a repeating of section 4.2.

The interaction on MPI in this case can be completed by Restconf protocol with topology and service YANG model. The interactions between PNC and network elements are vendor-specific.

#### 5. Implementation Details

The implementation and inter-op test is on-going. Once the result is available, this section will be updated.

The progress of future test work will be updated and available at the ACTN wiki page: <https://sites.google.com/site/openactn/>.

## 6. Future work

Inter-op test can be done either with emulation environment or physical devices in the lab. Some of the test related work in this draft will be done on emulations via remote labs of various vendors, or during IETF Hackathon activity. This will be a continuing activity with follow-up work in coming IETF meetings. More participants will be welcomed to join this effort to further promote IETF work to expedite SDN deployment.

## 7. Security Considerations

This document is an informational draft. When the models mentioned in this draft are implemented, detailed security consideration will be given in such work.

## 8. References

### 8.1. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
  
- [RFC6241] 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-Requirements] Y. Lee, et al., "ACTN Requirements for Abstraction and Control of TE Networks", draft-ietf-teas-actn-requirements, 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.
- [ACTN-Info] Y. Lee & S. Belotti, "Information Model for Abstraction and Control of TE Networks (ACTN)", draft-leebelotti-teas-actn-info, work in progress.
- [Transport-Service-Model] X. Zhang (Editor), "A Service YANG Model for Connection-oriented Transport Networks", draft-zhang-teas-transport-service-model, work in progress.
- [ACTN-YANG] Y. Lee, X. Zhang, D. Ceccarelli, B. Yoon, O. Gonzalez de Dios, J. Shin, Applicability of YANG models for Abstraction and Control of Traffic Engineered Networks, work in progress.
- [ACTN-PCE] D. Dhody, Y. Lee, D. Ceccarelli, Applicability of Path Computation Element (PCE) for Abstraction and Control of TE Networks (ACTN), work in progress.
- [PCE-LS] D. Dhody, Y. Lee, D. Ceccarelli, PCEP Extension for Distribution of Link-State and TE Information, work in progress.
- [PCE-Init] E. Crabbe, I. Minei, S. Sivabalan, R. Varga, PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model, work in progress.

## 9. Contributors' Addresses

Kun Xiang  
Huawei Technologies  
Email: xiangkun@huawei.com

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

Xin Liu  
Huawei Technologies  
Email liuxin12@huawei.com

Lei Wang  
China Communications Corporation  
Email wangleiyj@chinamobile.com

Weiqiang Cheng  
China Communications Corporation  
Email chengweiqiang@chinamobile.com

Liang Geng  
China Communications Corporation  
Email gengliang@chinamobile.com

Dong Wang  
China Communications Corporation  
Email wangdongyjy@chinamobile.com

Dhruv Dhody  
Huawei Technologies  
Email: dhruv.ietf@gmail.com

Satish Karunanithi  
Huawei Technologies  
Email: satishk@huawei.com

Wei Wang  
Beijing University of Post and Telecommunications  
Email: weiw@bupt.edu.cn

Yongli Zhao  
Beijing University of Post and Telecommunications  
Email: yonglizhao@bupt.edu.cn

Jie Zhang  
Beijing University of Post and Telecommunications  
Email: lgr24@bupt.edu.cn

## 10. Acknowledgment

TBD

### Authors' Address

Haomian Zheng  
Huawei Technologies  
Email: Zhenghaomian@huawei.com

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

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

Jong Yoon Shin  
SK Telecom

Email: jongyoon.shin@sk.com

Yunbin Xu  
China Academy of Information and Communication Technology  
Email: xuyunbin@mail.ritt.com.cn

Yunbo Li  
China Communications Corporation  
Email liyunbo@chinamobile.com

Yan Shi  
China Unicom  
Email shiyan49@chinaunicom.cn

Boyuan Yan  
Beijing University of Post and Telecommunications  
Email: yanboyuan@bupt.edu.cn

