RSVP-TE Extension for Beyond 100G Signal Types in G.709 Optical Transport Networks (OTNs)
draft-ali-ccamp-oducn-signal-type-00.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). 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.
Abstract

RFCs 4328 and 7139 provide signaling extensions in Resource ReserVation Protocol - Traffic Engineering (RSVP-TE) to control the full set of Optical Transport Network (OTN) features. However, these specifications do not cover the additional Optical channel Data Unit (ODU) containers defined in G.709/Y.1331 for ODUC1, ODUC2, ODUC3, ODUC4, ODUC5, ODUC6, ODUC7, ODUC8 and ODUC9. This document defines new Signal Types for these additional containers.

Conventions used in this document

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

Table of Contents

1. Introduction ................................................. 2
2. RSVP-TE extension for Beyond 100G Signal Types ............... 3
3. Security Considerations .......................................3
4. IANA Considerations ...........................................3
5. References ....................................................3
  5.1. Normative References ......................................3
  5.2. Informative References ....................................4

1. Introduction

[RFC7139] updates the portions of text related to the Optical channel Data Unit (ODU) described in [RFC4328] to provide extensions to Resource ReserVation Protocol - Traffic Engineering (RSVP-TE) to support control for [G.709-v3] in the OTN-TDM SENDERTSPEC and OTN-TDM FLOWSPEC objects. However, it does not specify Signal Types for the beyond 100G ODUCn containers defined in [G.709/Y.1331]. This document provides RSVP-TE signaling extensions to support ODUC1, ODUC2, ODUC3, ODUC4, ODUC5, ODUC6, ODUC7, ODUC8 and ODUC9 Signal Types.
2. RSVP-TE extension for Beyond 100G Signal Types

[RFC7139] defines the format of Traffic Parameters in OTN-TDM SENDER_TSPEC and OTN-TDM FLOWSPEC objects. These traffic parameters have a Signal Type field. This document defines the Signal Types for ODUC1, ODUC2, ODUC3, ODUC4, ODUC5, ODUC6, ODUC7, ODUC8 and ODUC9, as defined in the IANA Considerations section. They are allocated via the Specification Required policy added to the subregistry by [RFC7892].

3. Security Considerations

This document does not introduce any additional security issues beyond those identified in [RFC7139].

4. IANA Considerations

IANA maintains the "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Parameters" registry that contains the "OTN Signal Type" subregistry.

This document requests IANA to add the following signal types in the subregistry via the Specification Required policy [RFC5226]:

<table>
<thead>
<tr>
<th>Value</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA1</td>
<td>ODUC1</td>
<td>100Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA2</td>
<td>ODUC2</td>
<td>200Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA3</td>
<td>ODUC3</td>
<td>300Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA4</td>
<td>ODUC4</td>
<td>400Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA5</td>
<td>ODUC5</td>
<td>500Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA6</td>
<td>ODUC6</td>
<td>600Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA7</td>
<td>ODUC7</td>
<td>700Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA8</td>
<td>ODUC8</td>
<td>800Gbps OTN [G.709/Y.1331]</td>
</tr>
<tr>
<td>TBA9</td>
<td>ODUC9</td>
<td>900Gbps OTN [G.709/Y.1331]</td>
</tr>
</tbody>
</table>

These Signal Types are carried in the Traffic Parameters in OTN-TDM SENDER_TSPEC and OTN-TDM FLOWSPEC objects [RFC7139].

5. References

5.1. Normative References
5.2. Informative References


Authors' Addresses

Zafar Ali
Cisco Systems
Email: zali@cisco.com

Manoj Kumar
Cisco System
Email: manojk2@cisco.com

Akshaya Nadahalli
Cisco Systems
anadahal@cisco.com

Antonello Bonfanti
Cisco Systems
abonfant@cisco.com

Expires April 2017
Yang model for requesting Path Computation
draft-busibel-teas-yang-path-computation-00.txt

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt
Abstract

There are scenarios, typically in a hierarchical SDN context, in which an orchestrator may not have detailed information to be able to perform an end-to-end path computation and would need to request lower layer/domain controllers to calculate some (partial) feasible paths.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).

This document describes some use cases for a YANG model to request path computation.

Table of Contents

1. Introduction ......................................................3
2. Use Cases ........................................................4
   2.1. IP-Optical integration ....................................4
   2.1.1. Inter-layer path computation ..........................6
   2.1.2. Route Diverse IP Services .............................8
2.2. Multi-domain TE Networks ....................................8
1. Introduction

There are scenarios, typically in a hierarchical SDN context, in which an orchestrator may not have detailed information to be able to perform an end-to-end path computation and would need to request lower layer/domain controllers to calculate some (partial) feasible paths.

When we are thinking to this type of scenarios we have in mind specific level of interfaces on which this request can be applied.

We can reference ABNO Control Interface [RFC7491] in which an Application Service Coordinator can request ABNO controller to take in charge path calculation (see Figure 1 in the RFC) and/or ACTN [ACTN-frame], where controller hierarchy is defined, the need for path computation arises on both interfaces CMI (interface between Customer Network Controller (CNC) and Multi Domain Service Coordinator (MDSC)) and/or MPI (interface between MSDC-PNC). [ACTN-Info] describes an information model for the Path Computation request.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).
Path Computation Elements, Controllers and Orchestrators perform their operations based on Traffic Engineering Databases (TED). Such TEDs can be described, in a technology agnostic way, with the YANG Data Model for TE Topologies [TE-TOPO]. Furthermore, the technology specific details of the TED are modeled in the augmented TE topology models (e.g. [L1-TOPO] for Layer-1 ODU technologies).

The availability of such topology models allows providing the TED using YANG-based protocols (e.g., NETCONF or RESTCONF). Furthermore, it enables a PCE/Controller performing the necessary abstractions or modifications and offering this customized topology to another PCE/Controller or high level orchestrator.

The tunnels that can be provided over the networks described with the topology models can be also set-up, deleted and modified via YANG-based protocols (e.g., NETCONF or RESTCONF) using the TE-Tunnel Yang model [TE-TUNNEL].

This document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

2. Use Cases

This section presents different use cases, where an orchestrator needs to request underlying SDN controllers for path computation.

The presented uses cases have been grouped, depending on the different underlying topologies: a) IP-Optical integration; b) Multi-domain Traffic Engineered (TE) Networks; and c) Data center interconnections.

2.1. IP-Optical integration

In these use cases, an Optical domain is used to provide connectivity between IP routers which are connected with the Optical domains using access links (see Figure 1).
It is assumed that the Optical domain controller provides to the orchestrator an abstracted view of the Optical network. A possible abstraction shall be representing the optical domain as one "virtual node" with "virtual ports" connected to the access links.

The path computation request helps the orchestrator to know which are the real connections that can be provided at the optical domain.
2.1.1. Inter-layer path computation

In this use case, the orchestrator needs to setup an optimal path between two IP routers R1 and R2.

As depicted in Figure 2, the Orchestrator has only an "abstracted view" of the physical network, and it does not know the feasibility or the cost of the possible optical paths (e.g., VP1-VP4 and VP2-VP5), which depend from the current status of the physical resources within the optical network and on vendor-specific optical attributes.

The orchestrator can request the underlying Optical domain controller to compute a set of potential optimal paths, taking into account optical constraints. Then, based on its own constraints, policy and knowledge (e.g. cost of the access links), it can choose...
which one of these potential paths to use to setup the optimal e2e path crossing optical network.

Figure 3 - IP+Optical Path Computation Example

For example, in Figure 3, the Orchestrator can request the Optical domain controller to compute the paths between VP1-VP4 and VP2-VP5 and then decide to setup the optimal end-to-end path using the VP2-VP5 Optical path even this is not the optimal path from the Optical domain perspective.

Considering the dynamicity of the connectivity constraints of an Optical domain, it is possible that a path computed by the Optical domain controller when requested by the Orchestrator is no longer valid when the Orchestrator requests it to be setup up.

It is worth noting that with the approach proposed in this document, the likelihood for this issue to happen can be quite small since the time window between the path computation request and the path setup request should be quite short (especially if compared with the time that would be needed to update the information of a very detailed abstract connectivity matrix).

If this risk is still not acceptable, the Orchestrator may also optionally request the Optical domain controller not only to compute the path but also to keep track of its resources (e.g., these resources can be reserved to avoid being used by any other connection). In this case, some mechanism (e.g., a timeout) needs to be defined to avoid having stranded resources within the Optical domain.

These issues and solutions can be fine-tuned during the design of the YANG model for requesting Path Computation.
2.1.2. Route Diverse IP Services

This is for further study.

2.2. Multi-domain TE Networks

In this use case there are two TE domains which are interconnected together by multiple inter-domains links.

A possible example could be a multi-domain optical network.

---

Figure 4 - Multi-domain multi-link interconnection

In order to setup an end-to-end multi-domain TEpath (e.g., between nodes A and H), the orchestrator needs to know the feasibility or the cost of the possible TE paths within the two TE domains, which depend from the current status of the physical resources within each TE network. This is more challenging in case of optical networks because the optimal paths depend also on vendor-specific optical attributes (which may be different in the two domains if they are provided by different vendors).
In order to setup a multi-domain TE path (e.g., between nodes A and H), Orchestrator can request the TE domain controllers to compute a set of intra-domain optimal paths and take decisions based on the information received. For example:

- The Orchestrator asks TE domain controllers to provide set of paths between A-C, A-D, E-H and F-H
- TE domain controllers return a set of feasible paths with the associated costs: the path A-C is not part of this set (in optical networks, it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller)
- The Orchestrator will select the path A-D-F-H since it is the only feasible multi-domain path and then request the TE domain controllers to setup the A-D and F-H intra-domain paths
- If there are multiple feasible paths, the Orchestrator can select the optimal path knowing the cost of the intra-domain paths (provided by the TE domain controllers) and the cost of the inter-domain links (known by the Orchestrator)

This approach may have some scalability issues when the number of TE domains is quite big (e.g. 20).

In this case, it would be worthwhile using the abstract TE topology information provided by the domain controllers to limit the number of potential optimal end-to-end paths and then request path computation to fewer domain controllers in order to decide what the optimal path within this limited set is.

For more details, see section 3.3.

2.3. Data center interconnections

In these use cases, there is an TE domain which is used to provide connectivity between data centers which are connected with the TE domain using access links.
Figure 5 - Data Center Interconnection Use Case

In this use case, a virtual machine within Data Center 1 (DC1) needs to transfer data to another virtual machine that can reside either in DC2 or in DC3.

The optimal decision depends both on the cost of the TE path (DC1-DC2 or DC1-DC3) and of the computing power (data center resources) within DC2 or DC3.

The Cloud Orchestrator may not be able to make this decision because it has only an abstract view of the TE network (as in use case in 2.1).

The cloud orchestrator can request to the TE domain controller to compute the cost of the possible TE paths (e.g., DC1-DC2 and DC1-DC3) and to the DC controller to compute the cost of the computing power (DC resources) within DC2 and DC3 and then it can take the decision about the optimal solution based on this information and its policy.
3. Interactions with TE Topology

The use cases described in section 2 have been described assuming that the topology view exported by each underlying SDN controller to the orchestrator is aggregated using the "virtual node model", defined in [RFC7926].

TE Topology information, e.g., as provided by [TE-TOPO], could in theory be used by an underlying SDN controllers to provide TE information to the orchestrator thus allowing the Path Computation Element (PCE) within the Orchestrator to perform multi-domain path computation by its own, without requesting path computations to the underlying SDN controllers.

This section analyzes the need for an orchestrator to request underlying SDN controllers for path computation even in these scenarios as well as how the TE Topology information and the path computation can be complementary.

In nutshell, there is a scalability trade-off between providing all the TE information needed by the Orchestrator’s PCE to take optimal path computation decisions by its own versus requesting the Orchestrator to ask to too many underlying SDN Domain Controllers a set of feasible optimal intra-domain TE paths.

3.1. TE Topology Aggregation using the "virtual link model"

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export the whole TE domain as a single abstract TE node with a "detailed connectivity matrix", which extends the "connectivity matrix", defined in [RFC7446], with specific TE attributes (e.g., delay, SRLGs and summary TE metrics).

The information provided by the "detailed abstract connectivity matrix" would be equivalent to the information that should be provided by "virtual link model" as defined in [RFC 7926].

For example, in the IP-Optical integration use case, described in section 2.1, the Optical domain controller can make the information shown in Figure 3 available to the Orchestrator as part of the TE Topology information and the Orchestrator could use this information to calculate by its own the optimal path between routers R1 and R2, without requesting any additional information to the Optical Domain Controller.
However, there is a tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the Orchestrator’s PCE) and scalability to be considered when designing the amount of information to provide within the "detailed abstract connectivity matrix".

Figure 6 below shows another example, similar to Figure 3, where there are two possible Optical paths between VP1 and VP4 with different properties (e.g., available bandwidth and cost).

Figure 6 - IP+Optical Path Computation Example with multiple choices

Reporting all the information, as in Figure 6, using the "detailed abstract connectivity matrix", is quite challenging from a scalability perspective. The amount of this information is not just based on number of end points (which would scale as N-square), but also on many other parameters, including client rate, user constraints / policies for the service, e.g. max latency < N ms, max cost, etc., exclusion policies to route around busy links, min OSNR margin, max preFEC BER etc. All these constraints could be different based on connectivity requirements.

It is also worth noting that the "connectivity matrix" has been originally defined in WSON, [RFC7446] to report the connectivity constrains of a physical node within the WDM network: the information it contains is pretty "static" and therefore, once taken and stored in the TE data base, it can be always being considered valid and up-to-date in path computation request.

Using the "connectivity matrix" with an abstract node to abstract the information regarding the connectivity constraints of an Optical domain, would make this information more "dynamic" since the
Connectivity constraints of an Optical domain can change over time because some optical paths that are feasible at a given time may become unfeasible at a later time when e.g., another optical path is established. The information in the "detailed abstract connectivity matrix" is even more dynamic since the establishment of another optical path may change some of the parameters (e.g., delay or available bandwidth) in the "detailed abstract connectivity matrix" while not changing the feasibility of the path.

"Connectivity matrix" is sometimes confused with optical reach table that contain multiple (e.g. k-shortest) regen-free reachable paths for every A-Z node combination in the network. Optical reach tables can be calculated offline, utilizing vendor optical design and planning tools, and periodically uploaded to the Controller: these optical path reach tables are fairly static. However, to get the connectivity matrix, between any two sites, either a regen free path can be used, if one is available, or multiple regen free paths are concatenated to get from src to dest, which can be a very large combination. Additionally, when the optical path within optical domain needs to be computed, it can result in different paths based on input objective, constraints, and network conditions. In summary, even though "optical reachability table" is fairly static, which regen free paths to build the connectivity matrix between any source and destination is very dynamic, and is done using very sophisticated routing algorithms.

There is therefore the need to keep the information in the "connectivity matrix" updated which means that there another tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the Orchestrator’s PCE) and having up-to-date information. The more the information is provided and the longer it takes to keep it up-to-date which increases the likelihood that the Orchestrator’s PCE computes paths using not updated information.

It seems therefore quite challenging to have a "detailed abstract connectivity matrix" that provides accurate, scalable and updated information to allow the Orchestrator’s PCE to take optimal decisions by its own.

If the information in the "detailed abstract connectivity matrix" is not complete/accurate, we can have the following drawbacks considering for example the case in Figure 6:
If only the VP1-VP4 path with available bandwidth of 2 Gb/s and cost 50 is reported, the Orchestrator’s PCE will fail to compute a 5 Gb/s path between routers R1 and R2, although this would be feasible;

If only the VP1-VP4 path with available bandwidth of 10 Gb/s and cost 60 is reported, the Orchestrator’s PCE will compute, as optimal, the 1 Gb/s path between R1 and R2 going through the VP2-VP5 path within the Optical domain while the optimal path would actually be the one going thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

Instead, using the approach proposed in this document, the Orchestrator, when it needs to setup an end-to-end path, it can request the Optical domain controller to compute a set of optimal paths (e.g., for VP1-VP4 and VP2-VP5) and take decisions based on the information received:

When setting up a 5 Gb/s path between routers R1 and R2, the Optical domain controller may report only the VP1-VP4 path as the only feasible path: the Orchestrator can successfully setup the end-to-end path passing though this Optical path;

When setting up a 1 Gb/s path between routers R1 and R2, the Optical domain controller (knowing that the path requires only 1 Gb/s) can report both the VP1-VP4 path, with cost 50, and the VP2-VP5 path, with cost 65. The Orchestrator can then compute the optimal path which is passing thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

3.2. TE Topology Abstraction

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export an abstract TE Topology, composed by a set of TE nodes and TE links, which are abstracting the topology controlled by each domain controller.

Considering the example in Figure 4, the TE domain controller 1 can export a TE Topology encompassing the TE nodes A, B, C and D and the TE Link interconnecting them. In a similar way, TE domain controller 2 can export a TE Topology encompassing the TE nodes E, F, G and H and the TE Link interconnecting them.

In this example, for simplicity reasons, each abstract TE node maps with each physical node, but this is not necessary.
In order to setup a multi-domain TE path (e.g., between nodes A and H), the Orchestrator can compute by its own an optimal end-to-end path based on the abstract TE topology information provided by the domain controllers. For example:

- Orchestrator’s PCE, based on its own information, can compute the optimal multi-domain path being A-B-C-E-G-H, and then request the TE domain controllers to setup the A-B-C and E-G-H intra-domain paths.

- But, during path setup, the domain controller may find out that A-B-C intra-domain path is not feasible (as discussed in section 2.2, in optical networks it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller), while only the path A-B-D is feasible.

- So what the hierarchical controller computed is not good and need to re-start the path computation from scratch.

As discussed in section 3.1, providing more extensive abstract information from the TE domain controllers to the multi-domain Orchestrator may lead to scalability problems.

In a sense this is similar to the problem of routing and wavelength assignment within an Optical domain. It is possible to do first routing (step 1) and then wavelength assignment (step 2), but the chances of ending up with a good path is low. Alternatively, it is possible to do combined routing and wavelength assignment, which is known to be a more optimal and effective way for Optical path setup. Similarly, it is possible to first compute an abstract end-to-end path within the multi-domain Orchestrator (step 1) and then compute an intra-domain path within each Optical domain (step 2), but there are more chances not to find a path or to get a suboptimal path that performing per-domain path computation and then stitch them.

3.3. Complementary use of TE topology and path computation

As discussed in section 2.2, there are some scalability issues with path computation requests in a multi-domain TE network with many TE domains, in terms of the number of requests to send to the TE domain controllers. It would therefore be worthwhile using the TE topology information provided by the domain controllers to limit the number of requests.
An example can be described considering the multi-domain abstract topology shown in Figure 7. In this example, an end-to-end TE path between domains A and F needs to be setup. The transit domain should be selected between domains B, C, D and E.

---

Figure 7 - Multi-domain with many domains (Topology information)

The actual cost of each intra-domain path is not known a priori from the abstract topology information. The Orchestrator only knows, from the TE topology provided by the underlying domain controllers, the feasibility of some intra-domain paths and some upper-bound and/or lower-bound cost information. With this information, together with the cost of inter-domain links, the Orchestrator can understand by its own that:

- Domain B cannot be selected as the path connecting domains A and E is not feasible;
- Domain E cannot be selected as a transit domain since it is known from the abstract topology information provided by domain controllers that the cost of the multi-domain path A-E-F (which is 100, in the best case) will be always higher than the cost of the multi-domain paths A-D-F (which is 90, in the worst case) and A-E-F (which is 80, in the worst case).

Therefore, the Orchestrator can understand by its own that the optimal multi-domain path could be either A-D-F or A-E-F but it
cannot known which one of the two possible option actually provides the optimal end-to-end path.

The Orchestrator can therefore request path computation only to the TE domain controllers A, D, E and F (and not to all the possible TE domain controllers).

Figure 8 - Multi-domain with many domains (Path Computation information)

Based on these requests, the Orchestrator can know the actual cost of each intra-domain paths which belongs to potential optimal end-to-end paths, as shown in Figure 8, and then compute the optimal end-to-end path (e.g., A-D-F, having total cost of 50, instead of A-C-F having a total cost of 70).

4. Motivation for a YANG Model

4.1. Benefits of common data models

Path computation requests should be closely aligned with the YANG data models that provide (abstract) TE topology information, i.e., [TE-TOPO] as well as that are used to configure and manage TE Tunnels, i.e., [TE-TUNNEL]. Otherwise, an error-prone mapping or correlation of information would be required. For instance, there is benefit in using the same endpoint identifiers in path computation requests and in the topology modeling. Also, the attributes used in path computation constraints could use the same or similar data
models. As a result, there are many benefits in aligning path computation requests with YANG models for TE topology information and TE Tunnels configuration and management.

4.2. Benefits of a single interface

A typical use case for path computation requests is the interface between an orchestrator and a domain controller. The system integration effort is typically lower if a single, consistent interface is used between such systems, i.e., one data modeling language (i.e., YANG) and a common protocol (e.g., NETCONF or RESTCONF).

Practical benefits of using a single, consistent interface include:

1. Simple authentication and authorization: The interface between different components has to be secured. If different protocols have different security mechanisms, ensuring a common access control model may result in overhead. For instance, there may be a need to deal with different security mechanisms, e.g., different credentials or keys. This can result in increased integration effort.

2. Consistency: Keeping data consistent over multiple different interfaces or protocols is not trivial. For instance, the sequence of actions can matter in certain use cases, or transaction semantics could be desired. While ensuring consistency within one protocol can already be challenging, it is typically cumbersome to achieve that across different protocols.

3. Testing: System integration requires comprehensive testing, including corner cases. The more different technologies are involved, the more difficult it is to run comprehensive test cases and ensure proper integration.

4. Middle-box friendliness: Provider and consumer of path computation requests may be located in different networks, and middle-boxes such as firewalls, NATs, or load balancers may be deployed. In such environments it is simpler to deploy a single protocol. Also, it may be easier to debug connectivity problems.

5. Tooling reuse: Implementers may want to implement path computation requests with tools and libraries that already exist in controllers and/or orchestrators, e.g., leveraging the rapidly growing eco-system for YANG tooling.
4.3. Extensibility

Path computation is only a subset of the typical functionality of a controller. In many use cases, issuing path computation requests comes along with the need to access other functionality on the same system. In addition to obtaining TE topology, for instance also configuration of services (setup/modification/deletion) may be required, as well as:

1. Receiving notifications for topology changes as well as integration with fault management
2. Performance management such as retrieving monitoring and telemetry data
3. Service assurance, e.g., by triggering OAM functionality
4. Other fulfilment and provisioning actions beyond tunnels and services, such as changing QoS configurations

YANG is a very extensible and flexible data modeling language that can be used for all these use cases.

Adding support for path computation requests to YANG models would seamlessly complement with [TE-TOPO] and [TE-TUNNEL] in the use cases where YANG-based protocols (e.g., NETCONF or RESTCONF) are used.

5. Path Optimization Request

This is for further study

6. YANG Model for requesting Path Computation

Work on extending the TE Tunnel YANG model to support the need to request path computation has recently started also in the context of the [TE-TUNNEL] draft.

It is possible to request path computation by configuring a "compute-only" TE tunnel and retrieving the computed path(s) in the LSP(s) Record-Route Object (RRO) list as described in [TE-TUNNEL].

This is a stateful solution since the state of each created "compute-only" TE tunnel needs to be maintained and updated, when underlying network conditions change.

The need also for a stateless solution, based on an RPC, has been recognized.
The YANG model to support stateless RPC is for further study.

7. Security Considerations
   This is for further study

8. IANA Considerations
   This document requires no IANA actions.

9. References

9.1. Normative References


9.2. Informative References


10. Acknowledgments

The authors would like to thank Igor Bryskin and Xian Zhang for participating in discussions and providing valuable insights.

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

Dieter Beller
Nokia
Email: dieter.beller@nokia.com

Authors’ Addresses

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

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

Victor Lopez
Telefonica
Email: victor.lopezalvarez@telefonica.com

Oscar Gonzalez de Dios
Telefonica
Email: oscar.gonzalezdedios@telefonica.com

Anurag Sharma
Infinera
Email: AnSharma@infinera.com

Yan Shi
China Unicom
Email: shiyan49@chinaunicom.cn

Ricard Vilalta
CTTC
Email: ricard.vilalta@cttc.es
Yang model for requesting Path Computation
draft-busibel-teas-yang-path-computation-03.txt

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt
Abstract

There are scenarios, typically in a hierarchical SDN context, in which an orchestrator may not have detailed information to be able to perform an end-to-end path computation and would need to request lower layer/domain controllers to calculate some (partial) feasible paths.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).

This document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

This document also proposes a yang model for a stateless RPC which complements the stateful solution defined in [TE-TUNNEL].

Table of Contents

1. Introduction...................................................3
2. Use Cases......................................................4
1. Introduction

There are scenarios, typically in a hierarchical SDN context, in which an orchestrator may not have detailed information to be able to perform an end-to-end path computation and would need to request lower layer/domain controllers to calculate some (partial) feasible paths.

When we are thinking to this type of scenarios we have in mind specific level of interfaces on which this request can be applied.

We can reference ABNO Control Interface [RFC7491] in which an Application Service Coordinator can request ABNO controller to take in charge path calculation (see Figure 1 in the RFC) and/or ACTN [ACTN-frame], where controller hierarchy is defined, the need for path computation arises on both interfaces CMI (interface between Customer Network Controller (CNC) and Multi Domain Service Coordinator (MDSC)) and/or MPI (interface between MSDC-PNC).[ACTN-
Info] describes an information model for the Path Computation request.

Multiple protocol solutions can be used for communication between different controller hierarchical levels. This document assumes that the controllers are communicating using YANG-based protocols (e.g., NETCONF or RESTCONF).

Path Computation Elements, Controllers and Orchestrators perform their operations based on Traffic Engineering Databases (TED). Such TEDs can be described, in a technology agnostic way, with the YANG Data Model for TE Topologies [TE-TOPO]. Furthermore, the technology specific details of the TED are modeled in the augmented TE topology models (e.g. [L1-TOPO] for Layer-1 ODU technologies).

The availability of such topology models allows providing the TED using YANG-based protocols (e.g., NETCONF or RESTCONF). Furthermore, it enables a PCE/Controller performing the necessary abstractions or modifications and offering this customized topology to another PCE/Controller or high level orchestrator.

The tunnels that can be provided over the networks described with the topology models can be also set-up, deleted and modified via YANG-based protocols (e.g., NETCONF or RESTCONF) using the TE-Tunnel Yang model [TE-TUNNEL].

This document describes some use cases where a path computation request, via YANG-based protocols (e.g., NETCONF or RESTCONF), can be needed.

This document also proposes a yang model for a stateless RPC which complements the stateful solution defined in [TE-TUNNEL].

2. Use Cases

This section presents different use cases, where an orchestrator needs to request underlying SDN controllers for path computation.

The presented uses cases have been grouped, depending on the different underlying topologies: a) IP-Optical integration; b) Multi-domain Traffic Engineered (TE) Networks; and c) Data center interconnections.
2.1. IP-Optical integration

In these use cases, an Optical domain is used to provide connectivity between IP routers which are connected with the Optical domains using access links (see Figure 1).

It is assumed that the Optical domain controller provides to the orchestrator an abstracted view of the Optical network. A possible abstraction shall be representing the optical domain as one "virtual node" with "virtual ports" connected to the access links.

The path computation request helps the orchestrator to know which are the real connections that can be provided at the optical domain.
2.1.1. Inter-layer path computation

In this use case, the orchestrator needs to setup an optimal path between two IP routers R1 and R2.

As depicted in Figure 2, the Orchestrator has only an "abstracted view" of the physical network, and it does not know the feasibility or the cost of the possible optical paths (e.g., VP1-VP4 and VP2-VP5), which depend from the current status of the physical resources within the optical network and on vendor-specific optical attributes.

The orchestrator can request the underlying Optical domain controller to compute a set of potential optimal paths, taking into account optical constraints. Then, based on its own constraints, policy and knowledge (e.g. cost of the access links), it can choose...
which one of these potential paths to use to setup the optimal e2e path crossing optical network.

Figure 3 - IP+Optical Path Computation Example

For example, in Figure 3, the Orchestrator can request the Optical domain controller to compute the paths between VP1-VP4 and VP2-VP5 and then decide to setup the optimal end-to-end path using the VP2-VP5 Optical path even this is not the optimal path from the Optical domain perspective.

Considering the dynamicity of the connectivity constraints of an Optical domain, it is possible that a path computed by the Optical domain controller when requested by the Orchestrator is no longer valid when the Orchestrator requests it to be setup up.

It is worth noting that with the approach proposed in this document, the likelihood for this issue to happen can be quite small since the time window between the path computation request and the path setup request should be quite short (especially if compared with the time that would be needed to update the information of a very detailed abstract connectivity matrix).

If this risk is still not acceptable, the Orchestrator may also optionally request the Optical domain controller not only to compute the path but also to keep track of its resources (e.g., these resources can be reserved to avoid being used by any other connection). In this case, some mechanism (e.g., a timeout) needs to be defined to avoid having stranded resources within the Optical domain.

These issues and solutions can be fine-tuned during the design of the YANG model for requesting Path Computation.
2.1.2. Route Diverse IP Services

This is for further study.

2.2. Multi-domain TE Networks

In this use case there are two TE domains which are interconnected together by multiple inter-domains links.

A possible example could be a multi-domain optical network.

Figure 4 - Multi-domain multi-link interconnection

In order to setup an end-to-end multi-domain TE path (e.g., between nodes A and H), the orchestrator needs to know the feasibility or the cost of the possible TE paths within the two TE domains, which depend from the current status of the physical resources within each TE network. This is more challenging in case of optical networks because the optimal paths depend also on vendor-specific optical attributes (which may be different in the two domains if they are provided by different vendors).
In order to setup a multi-domain TE path (e.g., between nodes A and H), Orchestrator can request the TE domain controllers to compute a set of intra-domain optimal paths and take decisions based on the information received. For example:

- The Orchestrator asks TE domain controllers to provide set of paths between A-C, A-D, E-H and F-H
- TE domain controllers return a set of feasible paths with the associated costs: the path A-C is not part of this set (in optical networks, it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller)
- The Orchestrator will select the path A-D-F-H since it is the only feasible multi-domain path and then request the TE domain controllers to setup the A-D and F-H intra-domain paths
- If there are multiple feasible paths, the Orchestrator can select the optimal path knowing the cost of the intra-domain paths (provided by the TE domain controllers) and the cost of the inter-domain links (known by the Orchestrator)

This approach may have some scalability issues when the number of TE domains is quite big (e.g. 20).

In this case, it would be worthwhile using the abstract TE topology information provided by the domain controllers to limit the number of potential optimal end-to-end paths and then request path computation to fewer domain controllers in order to decide what the optimal path within this limited set is.

For more details, see section 3.3.

2.3. Data center interconnections

In these use case, there is an TE domain which is used to provide connectivity between data centers which are connected with the TE domain using access links.
Data Center Interconnection Use Case

(only in PDF version)

Figure 5 - Data Center Interconnection Use Case

In this use case, a virtual machine within Data Center 1 (DC1) needs to transfer data to another virtual machine that can reside either in DC2 or in DC3.

The optimal decision depends both on the cost of the TE path (DC1-DC2 or DC1-DC3) and of the computing power (data center resources) within DC2 or DC3.

The Cloud Orchestrator may not be able to make this decision because it has only an abstract view of the TE network (as in use case in 2.1).

The cloud orchestrator can request to the TE domain controller to compute the cost of the possible TE paths (e.g., DC1-DC2 and DC1-DC3) and to the DC controller to compute the cost of the computing power (DC resources) within DC2 and DC3 and then it can take the decision about the optimal solution based on this information and its policy.
3. Interactions with TE Topology

The use cases described in section 2 have been described assuming that the topology view exported by each underlying SDN controller to the orchestrator is aggregated using the "virtual node model", defined in [RFC7926].

TE Topology information, e.g., as provided by [TE-TOPO], could in theory be used by an underlying SDN controllers to provide TE information to the orchestrator thus allowing the Path Computation Element (PCE) within the Orchestrator to perform multi-domain path computation by its own, without requesting path computations to the underlying SDN controllers.

This section analyzes the need for an orchestrator to request underlying SDN controllers for path computation even in these scenarios as well as how the TE Topology information and the path computation can be complementary.

In nutshell, there is a scalability trade-off between providing all the TE information needed by the Orchestrator’s PCE to take optimal path computation decisions by its own versus requesting the Orchestrator to ask to too many underlying SDN Domain Controllers a set of feasible optimal intra-domain TE paths.

3.1. TE Topology Aggregation using the "virtual link model"

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export the whole TE domain as a single abstract TE node with a "detailed connectivity matrix", which extends the "connectivity matrix", defined in [RFC7446], with specific TE attributes (e.g., delay, SRLGs and summary TE metrics).

The information provided by the "detailed abstract connectivity matrix" would be equivalent to the information that should be provided by "virtual link model" as defined in [RFC7926].

For example, in the IP-Optical integration use case, described in section 2.1, the Optical domain controller can make the information shown in Figure 3 available to the Orchestrator as part of the TE Topology information and the Orchestrator could use this information to calculate by its own the optimal path between routers R1 and R2, without requesting any additional information to the Optical Domain Controller.
However, there is a tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the Orchestrator’s PCE) and scalability to be considered when designing the amount of information to provide within the "detailed abstract connectivity matrix".

Figure 6 below shows another example, similar to Figure 3, where there are two possible Optical paths between VP1 and VP4 with different properties (e.g., available bandwidth and cost).

---

Figure 6 - IP+Optical Path Computation Example with multiple choices

Reporting all the information, as in Figure 6, using the "detailed abstract connectivity matrix", is quite challenging from a scalability perspective. The amount of this information is not just based on number of end points (which would scale as N-square), but also on many other parameters, including client rate, user constraints / policies for the service, e.g. max latency < N ms, max cost, etc., exclusion policies to route around busy links, min OSNR margin, max preFEC BER etc. All these constraints could be different based on connectivity requirements.

In the following table, a list of the possible constraints, associated with their potential cardinality, is reported.

The maximum number of potential connections to be computed and reported is, in first approximation, the multiplication of all of them.

---

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>End points</td>
<td>(\frac{N(N-1)}{2}) if connections are bidirectional (OTN and WDM), (N(N-1)) for unidirectional connections.</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>In WDM networks, bandwidth values are expressed in GHz. On fixed-grid WDM networks, the central frequencies are on a 50GHz grid and the channel width of the transmitters are typically 50GHz such that each central frequency can be used, i.e., adjacent channels can be placed next to each other in terms of central frequencies. On flex-grid WDM networks, the central frequencies are on a 6.25GHz grid and the channel width of the transmitters can be multiples of 12.5GHz. For fixed-grid WDM networks typically there is only one possible bandwidth value (i.e., 50GHz) while for flex-grid WDM networks typically there are 4 possible bandwidth values (e.g., 37.5GHz, 50GHz, 62.5GHz, 75GHz). In OTN (ODU) networks, bandwidth values are expressed as pairs of ODU type and, in case of ODUflex, ODU rate in bytes/sec as described in section 5 of [RFC7139]. For &quot;fixed&quot; ODUk types, 6 possible bandwidth values are possible (i.e., ODU0, ODU1, ODU2, ODU2e, ODU3, ODU4). For ODUflex(GFP), up to 80 different bandwidth values can be specified, as defined in Table 7-8 of [ITU-T G.709-2016]. For other ODUflex types, like ODUflex(CBR), the number of possible bandwidth values depends on the rates of the clients that could be mapped over these ODUflex types, as shown in Table 7.2 of [ITU-T G.709-2016], which in theory could be a continuum of values. However, since different ODUflex bandwidths that use the same number of TSs on each link along the path are equivalent for path computation purposes, up to 120 different bandwidth ranges can be specified.</td>
</tr>
</tbody>
</table>
Ideas to reduce the number of ODUflex bandwidth values in the detailed connectivity matrix, to less than 100, are for further study.

Bandwidth specification for ODUCn is currently for further study but it is expected that other bandwidth values can be specified as integer multiples of 100Gb/s.

In IP we have bandwidth values in bytes/sec. In principle, this is a continuum of values, but in practice we can identify a set of bandwidth ranges, where any bandwidth value inside the same range produces the same path.
The number of such ranges is the cardinality, which depends on the topology, available bandwidth and status of the network. Simulations (Note: reference paper submitted for publication) show that values for medium size topologies (around 50-150 nodes) are in the range 4-7 (5 on average) for each end points couple.

**Metrics**
IGP, TE and hop number are the basic objective metrics defined so far. There are also the 2 objective functions defined in [RFC5541]: Minimum Load Path (MLP) and Maximum Residual Bandwidth Path (MBP). Assuming that one only metric or objective function can be optimized at once, the total cardinality here is 5.

With [PCEP-Service-Aware], a number of additional metrics are defined, including Path Delay metric, Path Delay Variation metric and Path Loss metric, both for point-to-point and point-to-multipoint paths. This increases the cardinality to 8.

**Bounds**
Each metric can be associated with a bound in order to find a path having a total value of that metric lower than the given bound. This has a potentially very high cardinality (as any value for the bound is allowed). In practice there is a maximum value of the bound (the one with the maximum value of the associated metric) which results always in the same path, and a range approach like for bandwidth in IP should produce also in this case the cardinality. Assuming to have a cardinality similar to the one of the bandwidth (let say 5 on average) we should have 6 (IGP, TE, hop, path delay, path delay variation and path loss; we don’t consider here the two
objective functions of [RFC5541] as they are conceived only for optimization)*5 = 30 cardinality.

**Priority**

We have 8 values for setup priority, which is used in path computation to route a path using free resources and, where no free resources are available, resources used by LSPs having a lower holding priority.

**Local prot**

It’s possible to ask for a local protected service, where all the links used by the path are protected with fast reroute (this is only for IP networks, but line protection schemas are available on the other technologies as well). This adds an alternative path computation, so the cardinality of this constraint is 2.

**Administrative Colors**

Administrative colors (aka affinities) are typically assigned to links but when topology abstraction is used affinity information can also appear in the detailed connectivity matrix.

There are 32 bits available for the affinities. Links can be tagged with any combination of these bits, and path computation can be constrained to include or exclude any or all of them. The relevant cardinality is 3 (include-any, exclude-any, include-all) times 2^32 possible values. However, the number of possible values used in real networks is quite small.

**Included Resources**

A path computation request can be associated to an ordered set of network resources (links, nodes) to be included along the computed path. This constraint would have a huge cardinality as in principle any combination of network resources is possible. However, as far as the Orchestrator doesn’t know details about the internal topology of the domain, it shouldn’t include this type of constraint at all (see more details below).

**Excluded Resources**

A path computation request can be associated to a set of network resources (links, nodes, SRLGs) to be excluded from the computed path. Like for included resources,
this constraint has a potentially very high cardinality, but, once again, it can’t be actually used by the Orchestrator, if it’s not aware of the domain topology (see more details below).

As discussed above, the Orchestrator can specify include or exclude resources depending on the abstract topology information that the domain controller exposes:

- In case the domain controller exposes the entire domain as a single abstract TE node with his own external terminations and connectivity matrix (whose size we are estimating), no other topological details are available, therefore the size of the connectivity matrix only depends on the combination of the constraints that the Orchestrator can use in a path computation request to the domain controller. These constraints cannot refer to any details of the internal topology of the domain, as those details are not known to the Orchestrator and so they do not impact size of connectivity matrix exported.

- Instead in case the domain controller exposes a topology including more than one abstract TE nodes and TE links, and their attributes (e.g. SRLGs, affinities for the links), the Orchestrator knows these details and therefore could compute a path across the domain referring to them in the constraints. The connectivity matrixes to be estimated here are the ones relevant to the abstract TE nodes exported to the Orchestrator. These connectivity matrixes and therefore theirs sizes, while cannot depend on the other abstract TE nodes and TE links, which are external to the given abstract node, could depend to SRLGs (and other attributes, like affinities) which could be present also in the portion of the topology represented by the abstract nodes, and therefore contribute to the size of the related connectivity matrix.

We also don’t consider here the possibility to ask for more than one path in diversity or for point-to-multi-point paths, which are for further study.

Considering for example an IP domain without considering SRLG and affinities, we have an estimated number of paths depending on these estimated cardinalities:

Endpoints = N*(N-1), Bandwidth = 5, Metrics = 6, Bounds = 20, Priority = 8, Local prot = 2
The number of paths to be pre-computed by each IP domain is therefore $24960 \times N(N-1)$ where $N$ is the number of domain access points.

This means that with just 4 access points we have nearly 300000 paths to compute, advertise and maintain (if a change happens in the domain, due to a fault, or just the deployment of new traffic, a substantial number of paths need to be recomputed and the relevant changes advertised to the upper controller).

This seems quite challenging. In fact, if we assume a mean length of 1K for the json describing a path (a quite conservative estimate), reporting 300000 paths means transferring and then parsing more than 300 Mbytes for each domain. If we assume that 20% (to be checked) of this paths change when a new deployment of traffic occurs, we have 60 Mbytes of transfer for each domain traversed by a new end-to-end path. If a network has, let say, 20 domains (we want to estimate the load for a non-trivial domain setup) in the beginning a total initial transfer of 6Gigs is needed, and eventually, assuming 4-5 domains are involved in mean during a path deployment we could have 240-300 Mbytes of changes advertised to the higher order controller.

Further bare-bone solutions can be investigated, removing some more options, if this is considered not acceptable; in conclusion, it seems that an approach based only on connectivity matrix is hardly feasible, and could be applicable only to small networks with a limited meshing degree between domains and renouncing to a number of path computation features.

It is also worth noting that the "connectivity matrix" has been originally defined in WSON, [RFC7446] to report the connectivity constrains of a physical node within the WDM network: the information it contains is pretty "static" and therefore, once taken and stored in the TE data base, it can be always being considered valid and up-to-date in path computation request.

Using the "connectivity matrix" with an abstract node to abstract the information regarding the connectivity constraints of an Optical domain, would make this information more "dynamic" since the connectivity constraints of an Optical domain can change over time because some optical paths that are feasible at a given time may become unfeasible at a later time when e.g., another optical path is established. The information in the "detailed abstract connectivity matrix" is even more dynamic since the establishment of another optical path may change some of the parameters (e.g., delay or
available bandwidth) in the "detailed abstract connectivity matrix" while not changing the feasibility of the path.

"Connectivity matrix" is sometimes confused with optical reach table that contain multiple (e.g. k-shortest) regen-free reachable paths for every A-Z node combination in the network. Optical reach tables can be calculated offline, utilizing vendor optical design and planning tools, and periodically uploaded to the Controller: these optical path reach tables are fairly static. However, to get the connectivity matrix, between any two sites, either a regen free path can be used, if one is available, or multiple regen free paths are concatenated to get from src to dest, which can be a very large combination. Additionally, when the optical path within optical domain needs to be computed, it can result in different paths based on input objective, constraints, and network conditions. In summary, even though "optical reachability table" is fairly static, which regen free paths to build the connectivity matrix between any source and destination is very dynamic, and is done using very sophisticated routing algorithms.

There is therefore the need to keep the information in the "connectivity matrix" updated which means that there another tradeoff between the accuracy (i.e., providing "all" the information that might be needed by the Orchestrator’s PCE) and having up-to-date information. The more the information is provided and the longer it takes to keep it up-to-date which increases the likelihood that the Orchestrator’s PCE computes paths using not updated information.

It seems therefore quite challenging to have a "detailed abstract connectivity matrix" that provides accurate, scalable and updated information to allow the Orchestrator’s PCE to take optimal decisions by its own.

If the information in the "detailed abstract connectivity matrix" is not complete/accurate, we can have the following drawbacks considering for example the case in Figure 6:

- If only the VP1-VP4 path with available bandwidth of 2 Gb/s and cost 50 is reported, the Orchestrator’s PCE will fail to compute a 5 Gb/s path between routers R1 and R2, although this would be feasible;
If only the VP1-VP4 path with available bandwidth of 10 Gb/s and cost 60 is reported, the Orchestrator’s PCE will compute, as optimal, the 1 Gb/s path between R1 and R2 going through the VP2-VP5 path within the Optical domain while the optimal path would actually be the one going thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

Instead, using the approach proposed in this document, the Orchestrator, when it needs to setup an end-to-end path, it can request the Optical domain controller to compute a set of optimal paths (e.g., for VP1-VP4 and VP2-VP5) and take decisions based on the information received:

- When setting up a 5 Gb/s path between routers R1 and R2, the Optical domain controller may report only the VP1-VP4 path as the only feasible path: the Orchestrator can successfully setup the end-to-end path passing though this Optical path;

- When setting up a 1 Gb/s path between routers R1 and R2, the Optical domain controller (knowing that the path requires only 1 Gb/s) can report both the VP1-VP4 path, with cost 50, and the VP2-VP5 path, with cost 65. The Orchestrator can then compute the optimal path which is passing thought the VP1-VP4 sub-path (with cost 50) within the Optical domain.

### 3.2. TE Topology Abstraction

Using the TE Topology model, as defined in [TE-TOPO], the underlying SDN controller can export an abstract TE Topology, composed by a set of TE nodes and TE links, which are abstracting the topology controlled by each domain controller.

Considering the example in Figure 4, the TE domain controller 1 can export a TE Topology encompassing the TE nodes A, B, C and D and the TE Link interconnecting them. In a similar way, TE domain controller 2 can export a TE Topology encompassing the TE nodes E, F, G and H and the TE Link interconnecting them.

In this example, for simplicity reasons, each abstract TE node maps with each physical node, but this is not necessary.

In order to setup a multi-domain TE path (e.g., between nodes A and H), the Orchestrator can compute by its own an optimal end-to-end path based on the abstract TE topology information provided by the domain controllers. For example:
o Orchestrator’s PCE, based on its own information, can compute the optimal multi-domain path being A-B-C-E-G-H, and then request the TE domain controllers to setup the A-B-C and E-G-H intra-domain paths

o But, during path setup, the domain controller may find out that A-B-C intra-domain path is not feasible (as discussed in section 2.2, in optical networks it is typical to have some paths not being feasible due to optical constraints that are known only by the optical domain controller), while only the path A-B-D is feasible

o So what the hierarchical controller computed is not good and need to re-start the path computation from scratch

As discussed in section 3.1, providing more extensive abstract information from the TE domain controllers to the multi-domain Orchestrator may lead to scalability problems.

In a sense this is similar to the problem of routing and wavelength assignment within an Optical domain. It is possible to do first routing (step 1) and then wavelength assignment (step 2), but the chances of ending up with a good path is low. Alternatively, it is possible to do combined routing and wavelength assignment, which is known to be a more optimal and effective way for Optical path setup. Similarly, it is possible to first compute an abstract end-to-end path within the multi-domain Orchestrator (step 1) and then compute an intra-domain path within each Optical domain (step 2), but there are more chances not to find a path or to get a suboptimal path that performing per-domain path computation and then stitch them.

3.3. Complementary use of TE topology and path computation

As discussed in section 2.2, there are some scalability issues with path computation requests in a multi-domain TE network with many TE domains, in terms of the number of requests to send to the TE domain controllers. It would therefore be worthwhile using the TE topology information provided by the domain controllers to limit the number of requests.

An example can be described considering the multi-domain abstract topology shown in Figure 7. In this example, an end-to-end TE path between domains A and F needs to be setup. The transit domain should be selected between domains B, C, D and E.
The actual cost of each intra-domain path is not known a priori from the abstract topology information. The Orchestrator only knows, from the TE topology provided by the underlying domain controllers, the feasibility of some intra-domain paths and some upper-bound and/or lower-bound cost information. With this information, together with the cost of inter-domain links, the Orchestrator can understand by its own that:

- Domain B cannot be selected as the path connecting domains A and E is not feasible;
- Domain E cannot be selected as a transit domain since it is know from the abstract topology information provided by domain controllers that the cost of the multi-domain path A-E-F (which is 100, in the best case) will be always be higher than the cost of the multi-domain paths A-D-F (which is 90, in the worst case) and A-E-F (which is 80, in the worst case)

Therefore, the Orchestrator can understand by its own that the optimal multi-domain path could be either A-D-F or A-E-F but it cannot known which one of the two possible option actually provides the optimal end-to-end path.

The Orchestrator can therefore request path computation only to the TE domain controllers A, D, E and F (and not to all the possible TE domain controllers).
Based on these requests, the Orchestrator can know the actual cost of each intra-domain paths which belongs to potential optimal end-to-end paths, as shown in Figure 8, and then compute the optimal end-to-end path (e.g., A-D-F, having total cost of 50, instead of A-C-F having a total cost of 70).

4. Motivation for a YANG Model

4.1. Benefits of common data models

Path computation requests should be closely aligned with the YANG data models that provide (abstract) TE topology information, i.e., [TE-TOPO] as well as that are used to configure and manage TE Tunnels, i.e., [TE-TUNNEL]. Otherwise, an error-prone mapping or correlation of information would be required. For instance, there is benefit in using the same endpoint identifiers in path computation requests and in the topology modeling. Also, the attributes used in path computation constraints could use the same or similar data models. As a result, there are many benefits in aligning path computation requests with YANG models for TE topology information and TE Tunnels configuration and management.
4.2. Benefits of a single interface

A typical use case for path computation requests is the interface between an orchestrator and a domain controller. The system integration effort is typically lower if a single, consistent interface is used between such systems, i.e., one data modeling language (i.e., YANG) and a common protocol (e.g., NETCONF or RESTCONF).

Practical benefits of using a single, consistent interface include:

1. Simple authentication and authorization: The interface between different components has to be secured. If different protocols have different security mechanisms, ensuring a common access control model may result in overhead. For instance, there may be a need to deal with different security mechanisms, e.g., different credentials or keys. This can result in increased integration effort.
2. Consistency: Keeping data consistent over multiple different interfaces or protocols is not trivial. For instance, the sequence of actions can matter in certain use cases, or transaction semantics could be desired. While ensuring consistency within one protocol can already be challenging, it is typically cumbersome to achieve that across different protocols.
3. Testing: System integration requires comprehensive testing, including corner cases. The more different technologies are involved, the more difficult it is to run comprehensive test cases and ensure proper integration.
4. Middle-box friendliness: Provider and consumer of path computation requests may be located in different networks, and middle-boxes such as firewalls, NATs, or load balancers may be deployed. In such environments it is simpler to deploy a single protocol. Also, it may be easier to debug connectivity problems.
5. Tooling reuse: Implementers may want to implement path computation requests with tools and libraries that already exist in controllers and/or orchestrators, e.g., leveraging the rapidly growing eco-system for YANG tooling.

4.3. Extensibility

Path computation is only a subset of the typical functionality of a controller. In many use cases, issuing path computation requests comes along with the need to access other functionality on the same
system. In addition to obtaining TE topology, for instance also configuration of services (setup/modification/deletion) may be required, as well as:

1. Receiving notifications for topology changes as well as integration with fault management
2. Performance management such as retrieving monitoring and telemetry data
3. Service assurance, e.g., by triggering OAM functionality
4. Other fulfilment and provisioning actions beyond tunnels and services, such as changing QoS configurations

YANG is a very extensible and flexible data modeling language that can be used for all these use cases.

Adding support for path computation requests to YANG models would seamlessly complement with [TE-TOPO] and [TE-TUNNEL] in the use cases where YANG-based protocols (e.g., NETCONF or RESTCONF) are used.

5. Path Computation for multiple LSPs

There are use cases, where path computation is required for multiple Traffic Engineering Label Switched Paths (TE LSPs) through a network or through a network domain. It may be advantageous to request the new paths for a set of LSPs in one single path computation request [RFC5440] that also includes information regarding the desired objective function, see [RFC5541].

In the context of abstraction and control of TE networks (ACTN), as defined in [ACTN-Frame], when a MDSC receives a virtual network (VN) request from a CNC, the MDSC needs to perform path computation for multiple LSPs as a typical VN is constructed by a set of multiple paths also called end-to-end tunnels. The MDSC may send a single path computation request to the PNC for multiple LSPs, i.e. between the VN end points (access points in ACTN terminology).

In a more general context, when a MDSC needs to send multiple path provisioning requests to the PNC, the MDSC may also group these path provisioning requests together and send them in a single message to the PNC instead of sending separate requests for each path.
6. YANG Model for requesting Path Computation

The TE Tunnel YANG model has been extended to support the need to request path computation.

It is possible to request path computation by configuring a "compute-only" TE tunnel and retrieving the computed path(s) in the LSP(s) Record-Route Object (RRO) list as described in section 3.3.1 of [TE-TUNNEL].

This is a stateful solution since the state of each created "compute-only" TE tunnel needs to be maintained and updated, when underlying network conditions change.

The need also for a stateless solution, based on an RPC, has been recognized, as outlined in section 6.1.

A proposal for a stateless RPC to request path computation is provided in section 6.2.

6.1. Stateless and Stateful Path Computation

It is very useful to provide options for both stateless and stateful path computation mechanisms. It is suggested to use stateless mechanisms as much as possible and to rely on stateful path computation when really needed.

Stateless RPC allows requesting path computation using a simple atomic operation and it is the natural option/choice, especially with stateless PCE.

Since the operation is stateless, there is no guarantee that the returned path would still be available when path setup is requested: this is not a major issue in case the time between path computation and path setup is short.

The RPC response must be provided synchronously and, if collaborative computations are time consuming, it may not be possible to immediate reply to client.

In this case, the client can define a maximum time it can wait for the reply, such that if the computation does not complete in time, the server will abort the path computation and reply to the client with an error. It may be possible that the server has tighter timing
Internet-DraftYang model for requesting Path Computation      June 2017

constraints than the client: in this case the path computation is
aborted earlier than the time specified by the client.

Note - The RPC response issue (slow RPC server) is not specific to
the path computation RPC case so, it may be worthwhile, evaluating
whether a more generic solution applicable to any YANG RPC can be
used instead.

In case the stateless solution is not sufficient, a stateful
solution, based on "compute-only" TE tunnel, could be used to
support asynchronous operations and/or to get notifications in case
the computed path has been changed.

It is worth noting that also the stateful solution, although
increasing the likelihood that the computed path is available at
path setup, it does not guaranteed that because notifications may
not be reliable or delivered on time.

The stateful path computation has also the following drawbacks:

- Several messages required for any path computation
- Requires persistent storage in the provider controller
- Need for garbage collection for stranded paths
- Process burden to detect changes on the computed paths in order
to provide notifications update

6.2. YANG model for stateless TE path computation

6.2.1. YANG Tree

Figure 9 below shows the tree diagram of the YANG model defined in
module ietf-te-path-computation.yang.

module: ietf-te-path-computation
  +--rw paths
    |   +--ro path* [path-id]
    |     +--ro _telink* [link-ref]
    |     +--ro link-ref ->
    /nd:networks/network[nd:network-id=current()]/../network-ref]/lnk:link/link-id
Internet-Draft Yang model for requesting Path Computation May 2017


---ro path-constraints
   ---ro path-metric-bound* [metric-type]
      | ---ro metric-type identityref
      | ---ro upper-bound? uint64
   ---ro topology-id? te-types:te-topology-id
   ---ro ignore-overload? boolean

---ro bandwidth-generic
   ---ro te-bandwidth
      ---ro (technology)?
         +-:(psc)
         |   ---ro psc? rt-types:bandwidth-ieee-float32
         |   ---ro rate-type identityref
         |   ---ro counter? uint16
         |   ---ro (otn)
         |     ---ro rate-type identityref
         |     ---ro counter? uint16
         |     ---ro (lsc)
         |     ---ro (generic)
   |   ---ro disjointness? te-types:te-path-disjointness?

---ro setup-priority? uint8
---ro hold-priority? uint8
---ro signaling-type? identityref

---ro path-affinities
   ---ro constraint* [usage]
      | ---ro usage identityref
      | ---ro value? admin-groups

---ro path-srlgs
   ---ro usage? identityref
   ---ro values* srlg

---ro path-id yang-types:uuid

---ro pathComputationService
   ---ro _path-ref* -> /paths/path/path-id
+-ro path-affinities
  +--ro constraint* [usage]
    +--ro usage    identityref
    +--ro value?   admin-groups
  +--ro path-srlgs
    +--ro usage?   identityref
    +--ro values*  srlg
  +--ro optimizations
    +--ro (algorithm)?
      +--:(metric) {path-optimization-metric}?
        +--ro optimization-metric* [metric-type]
          +--ro metric-type identityref
          +--ro weight?   uint8
        +--ro tiebreakers
          +--ro tiebreaker* [tiebreaker-type]
            +--ro tiebreaker-type identityref
        +--:(objective-function) {path-optimization-objective-function}?
          +--ro objective-function
            +--ro objective-function-type? identityref
augment /te:tunnels-rpc/te:input/te:tunnel-info:
  +----- request-list* [request-id-number]
    +----- request-id-number   uint32
    +----- servicePort*
      +----- source?    inet:ip-address
      +----- destination? inet:ip-address
      +----- src-tp-id?   binary
      +----- dst-tp-id?   binary
      +----- bidirectional
        +----- id?            uint16
        +----- source?       inet:ip-address
        +----- global-source? inet:ip-address
        +----- type?         identityref
        +----- provisioning?  identityref
    +----- path-constraints
      +----- path-metric-bound* [metric-type]
        +----- metric-type identityref
        +----- upper-bound?  uint64
+++: (objective-function) {path-optimization-objective-function}?
  +++: objective-function
    +++: objective-function-type? identityref
  +++: synchronization* [synchronization-index]
    +++: synchronization-index uint32
  +++: svec
    +++: relaxable? boolean
    +++: link-diverse? boolean
    +++: node-diverse? boolean
    +++: srlg-diverse? boolean
    +++: request-id-number* uint32
  +++: path-constraints
    +++: path-metric-bound* [metric-type]
      +++: metric-type identityref
      +++: upper-bound? uint64
    +++: topology-id? te-types:te-topology-id
    +++: ignore-overload? boolean
  +++: bandwidth-generic
    +++: (technology)?
      +++: (psc)
        +++: psc? rt-types:bandwidth-ieee-float32
        +++: (otn)
          +++: otn* [rate-type]
            +++: rate-type identityref
            +++: counter? uint16
        +++: (lsc)
          +++: wdm* [spectrum slot]
            +++: spectrum identityref
            +++: slot int16
            +++: width? uint16
      +++: (generic)
        +++: generic? te-bandwidth
    +++: disjointness? te-types:te-path-disjointness
    +++: setup-priority? uint8
    +++: hold-priority? uint8
    +++: signaling-type? identityref
++---- path-affinities
| ++---- constraint* [usage]
| | ++---- usage  identityref
| | ++---- value?  admin-groups
++---- path-srlgs
++---- usage?  identityref
++---- values*  srlg
augment /te:tunnels-rpc/te:output/te:result:
++--ro response* [response-index]
| ++--ro response-index  uint32
++--ro (response-type)?
| ++--:(no-path-case)
| | ++--ro no-path
| ++--:(path-case)
++--ro pathCompService
++--ro_path-ref*  -> /paths/path/path-id
++--ro_servicePort
| ++--ro source?  inet:ip-address
| ++--ro destination?  inet:ip-address
| ++--ro src-tp-id?  binary
| ++--ro dst-tp-id?  binary
++--ro bidirectional
| ++--ro association
| | ++--ro id?  uint16
| | ++--ro source?  inet:ip-address
| | ++--ro global-source?  inet:ip-address
| | ++--ro type?  identityref
| | ++--ro provisioning?  identityref
++--ro path-constraints
++--ro path-metric-bound* [metric-type]
| | ++--ro metric-type  identityref
| | ++--ro upper-bound?  uint64
| | ++--ro topology-id?  te-types:te-topology-id
| | ++--ro ignore-overload?  boolean
| | ++--ro bandwidth-generic
| | | ++--ro te-bandwidth
| | | | ++--ro (technology)?
| | | | | ++--:(psc)
6.2.2. YANG Module

<CODE BEGINS>file "ietf-te-path-computation.yang "
module ietf-te-path-computation {
  yang-version 1.1;
  // replace with IANA namespace when assigned
  prefix "tepc";

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

  import ietf-yang-types {
    prefix "yang-types";
  }

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

  import ietf-te {
    prefix "te";
  }

  import ietf-te-types {
    prefix "te-types";
  }

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

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

feature stateless-path-computation {
  description "This feature indicates that the system supports stateless path computation.";
}

/* Groupings */

grouping Path {
  list _telink {
    key 'link-ref';
    config false;
    uses nt:link-ref;
    description "List of telink refs.";
  }
  uses te-types:generic-path-constraints;
}
leaf path-id {
    type yang-types:uuid;
    config false;
    description "path-id ref.";
}
description "Path is described by an ordered list of TE Links.";

grouping PathCompServicePort {
    leaf source {
        type inet:ip-address;
        description "TE tunnel source address.";
    }
    leaf destination {
        type inet:ip-address;
        description "P2P tunnel destination address";
    }
    leaf src-tp-id {
        type binary;
        description "TE tunnel source termination point identifier.";
    }
    leaf dst-tp-id {
        type binary;
        description "TE tunnel destination termination point identifier.";
    }
    uses te:bidir-assoc-properties;
    description "Path Computation Service Port grouping.";
}

grouping PathComputationService {
    leaf-list _path-ref {
        type leafref {
            path '/paths/path/path-id';
        }
        config false;
        description "List of previously computed path references.";
    }
}
container _servicePort {
    uses PathCompServicePort;
    description "Path Computation Service Port.";
}
uses te-types:generic-path-constraints;
uses te-types:generic-path-optimization;

description "Path computation service.";
}

grouping synchronization-info {
    description "Information for sync";
    list synchronization {
        key "synchronization-index";
        description "sync list";
        leaf synchronization-index {
            type uint32;
            description "index";
        }
    }
    container svec {
        description "Synchronization VECTor";
        leaf relaxable {
            type boolean;
            default true;
            description "If this leaf is true, path computation process is free to ignore svec content. otherwise it must take into account this svec.";
        }
        leaf link-diverse {
            type boolean;
            default false;
            description "link-diverse";
        }
        leaf node-diverse {
        }
    }
}
leaf srlg-diverse {
  type boolean;
  default false;
  description "srlg-diverse";
}
leaf-list request-id-number {
  type uint32;
  description "This list reports the set of M path computation requests that must be synchronized.";
}
  uses te-types:generic-path-constraints;
}

grouping no-path-info {
  description "no-path-info";
  container no-path {
    description "no-path container";
  }
}

/* * Root container */
container paths {
  list path {
    key "path-id";
    config false;
    uses Path;
    description "List of previous computed paths.";
  }
  description "Root container for path-computation";
}
container pathComputationService {
    config false;
    uses PathComputationService;
    description "Service for computing paths.";
}

/**
 * AUGMENTS TO TE RPC
 */
augment "/te:tunnels-rpc/te:input/te:tunnel-info" {
    description "statelessComputeP2PPath input";
    list request-list {
        key "request-id-number";
        description "request-list";
        leaf request-id-number {
            type uint32;
            mandatory true;
            description "Each path computation request is uniquely identified by the request-id-number. It must be present also in rpcs.";
        }
    }
    list servicePort {
        min-elements 1;
        uses PathCompServicePort;
        description "List of service ports.";
    }
    uses te-types:generic-path-constraints;
    uses te-types:generic-path-optimization;
}

uses synchronization-info;

augment "/te:tunnels-rpc/te:output/te:result" {

description "statelessComputeP2PPath output";
list response {
  key response-index;
  config false;
  description "response";
  leaf response-index {
    type uint32;
    description "The list key that has to reuse request-id-number.";
  }
  choice response-type {
    config false;
    description "response-type";
    case no-path-case {
      uses no-path-info;
    }
    case path-case {
      container pathCompService {
        uses PathComputationService;
        description "Path computation service.";
      }
    }
  }
}

Figure 10 - TE path computation YANG module

7. Security Considerations

This document describes use cases of requesting Path Computation using YANG models, which could be used at the ABNO Control Interface [RFC7491] and/or between controllers in ACTN [ACTN-frame]. As such, it does not introduce any new security considerations compared to the ones related to YANG specification, ABNO specification and ACTN Framework defined in [RFC6020], [RFC7950], [RFC7491] and [ACTN-frame].
This document also defines common data types using the YANG data modeling language. The definitions themselves have no security impact on the Internet, but the usage of these definitions in concrete YANG modules might have. The security considerations spelled out in the YANG specification [RFC6020] apply for this document as well.

8. IANA Considerations

This section is for further study: to be completed when the YANG model is more stable.

9. References

9.1. Normative References


9.2. Informative References


10. Acknowledgments

The authors would like to thank Igor Bryskin and Xian Zhang for participating in discussions and providing valuable insights.

The authors would like to thank the authors of the TE Tunnel YANG model [TE-TUNNEL], in particular Igor Bryskin, Vishnu Pavan Beeram, Tarek Saad and Xufeng Liu, for their inputs to the discussions and support in having consistency between the Path Computation and TE Tunnel YANG models.

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

Dieter Beller
Nokia
Email: dieter.beller@nokia.com

Gianmarco Bruno
Ericsson
Email: gianmarco.bruno@ericsson.com

Francesco Lazzeri
Ericsson
Email: francesco.lazzeri@ericsson.com

Young Lee
Huawei
Email: leeyoung@huawei.com

Carlo Perocchio
Ericsson
Email: carlo.perocchio@ericsson.com

Authors’ Addresses

Italo Busi (Editor)
Huawei
Email: italo.busi@huawei.com

Sergio Belotti (Editor)
Nokia
Email: sergio.belotti@nokia.com

Victor Lopez
Telefonica
Email: victor.lopezalvarez@telefonica.com
Abstract

This memo defines a Yang model related to the Optical Transceiver optical parameters characterising the 100G and above interfaces. 100G and above Transceivers support coherent transmission, different modulation format, multiple FEC algorithms not yet specified by ITU-T G.698.2 [ITU.G698.2] or any other ITU-T recommendation. The use cases and the state of the Coherent transceivers is well describe in draft-many-coherent-DWDM-if-control.

The Yang model defined in this memo can be used for Optical Parameters monitoring and/or configuration of the endpoints of the multi-vendor IaDI optical link.
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. The Internet-Standard Management Framework .................. 4
3. Conventions ................................................... 4
4. Overview ....................................................... 4
   4.1. Optical Parameters Description .......................... 5
      4.1.1. Table of Application Codes .......................... 6
      4.1.2. Rs-Ss Configuration and operating parameters ....... 6
   4.2. Parameters at Ss ......................................... 8
   4.3. Interface at point Rs .................................... 8
      4.3.1. Mandatory parameters ............................... 9
      4.3.2. Optional parameters ............................... 9
      4.3.3. Optical path from point Ss to Rs ................... 9
   4.4. Use Cases ................................................ 9
   4.5. Optical Interface for external transponder in a WDM network ........................................... 10
5. Structure of the Yang Module ................................ 10
6. Yang Module .................................................. 11
7. Security Considerations ...................................... 18
8. IANA Considerations .......................................... 18
9. Acknowledgements ............................................. 18
10. Contributors ................................................ 19
11. References .................................................. 19
   11.1. Normative References .................................. 19
   11.2. Informative References ................................ 22
Appendix A. Change Log .......................................... 22
Appendix B. Open Issues ........................................ 22
Authors’ Addresses ............................................. 22
1. Introduction

This memo defines a Yang model that translates and obsolete the SNMP mib module defined in draft-galikunze-ccamp-dwdm-if-snmp-mib for managing single channel optical interface parameters of DWDM applications, using the approach specified in G.698.2. This model is to support the optical parameters specified in ITU-T G.698.2 [ITU.G698.2], plus some parameters related to full coherent transmission and not yet specified by ITU-T like modulation format, finer Grid provisioning, multiple carrier, etc. The application identifiers specified in ITU-T G.874.1 [ITU.G874.1] and the Optical Power at Transmitter and Receiver side. Note that G.874.1 encompasses vendor-specific codes, which if used would make the interface a single vendor IaDI and could still be managed.

[Editor’s note: In G.698.2 this corresponds to the optical path from point S to R; network media channel is also used and explained in draft-ietf-ccamp-flexi-grid-fwk-02]

Management will be performed at the edges of the network media channel (i.e., at the transmitters and receivers attached to the S and R reference points respectively) for the relevant parameters specified in G.698.2 [ITU.G698.2], G.798 [ITU.G798], G.874 [ITU.G874], and the performance parameters specified in G.7710/Y.1701 [ITU-T G.7710] and G.874.1 [ITU.G874.1].

G.698.2 [ITU.G698.2] is primarily intended for metro applications that include optical amplifiers. Applications are defined in G.698.2 [ITU.G698.2] using optical interface parameters at the single-channel connection points between optical transmitters and the optical multiplexer, as well as between optical receivers and the optical demultiplexer in the DWDM system. This Recommendation uses a methodology which does not explicitly specify the details of the optical network between reference point Ss and Rs, e.g., the passive and active elements or details of the design. The Recommendation currently includes unidirectional DWDM applications at 2.5 and 10 Gbit/s (with 100 GHz and 50 GHz channel frequency spacing). Work is still under way for 40, 100 and Higher Gbit/s interfaces. There is possibility for extensions to a lower channel frequency spacing. This document specifically refers also to the “application code” defined in the G.698.2 [ITU.G698.2] and included in the Application Identifier defined in G.874.1 [ITU.G874.1] and G.872 [ITU.G872], plus a few optical parameters not included in the G.698.2 application code specification.

This draft refers and supports the draft-ietf-ccamp-dwdm-if-mng-ctrl-fwk and draft-many-coherent-DWDM-if-control.
The building of a yang model describing and extending the optical parameters defined in G.698.2 [ITU.G698.2], and reflected in G.874.1 [ITU.G874.1], allows the different vendors and operator to retrieve, provision and exchange information across the G.698.2 multi-vendor IaDI in a standardised way. In addition to the parameters specified in ITU recommendations the Yang models support also the "vendor specific application identifier", the Tx and Rx power at the Ss and Rs points and the channel frequency and the detailed parameters described in G.698.2 extending them to the new 100G and higher coherent interfaces.

The Yang Model, reporting the Optical parameters and their values, characterizes the features and the performances of the optical components and allow a reliable link design in case of multi vendor optical networks.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

This memo specifies a Yang model for optical interfaces.

3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] In the description of OIDs the convention: Set (S) Get (G) and Trap (T) conventions will describe the action allowed by the parameter.

4. Overview
Figure 1 shows a set of reference points, for single-channel connection between transmitters (Tx) and receivers (Rx). Here the DWDM network elements include an OM and an OD (which are used as a pair with the opposing element), one or more optical amplifiers and may also include one or more OADMs.

Ss = reference point at the DWDM network element tributary output
Rs = reference point at the DWDM network element tributary input
Lx = Lambda x
OM = Optical Mux
OD = Optical Demux
OADM = Optical Add Drop Mux

from Fig. 5.1/G.698.2

Figure 1: External transponder in WDM networks

4.1. Optical Parameters Description

The link between the external transponders through a WDM network media channels are managed at the edges, i.e. at the transmitters (Tx) and receivers (Rx) attached to the S and R reference points respectively. The set of parameters that could be managed are defined by the "application code" notation

The definitions of the optical parameters are provided below to increase the readability of the document, where the definition is
ended by (R) the parameter can be retrieve with a read, when (W) it
can be provisioned by a write, (R,W) can be either read or written.

4.1.1. Table of Application Codes

This table has a list of Application codes supported by this
interface at point R are defined in G.698.2.

Application code Identifier:
The Identifier for the Application code.

Application code Type:
This parameter indicates the transceiver type of application
code at Ss and Rs as defined in [ITU.G874.1], that is used by
this interface. Standard = 0, PROPRIETARY = 1
If Proprietary the first 6 octets of the printable string will
be the OUI (organizationally unique identifier) assigned to
the vendor whose implementation generated the Application
Identifier Code.

Application code:
This is the application code that is defined in G.698.2 or the
vendor generated code which has the OUI.

Number of Single-channel application codes Supported:
This parameter indicates the number of Single-channel
application codes supported by this interface.

Application code Length:
The number of octets in the Application Code.

4.1.2. Rs-Ss Configuration and operating parameters

The Rs-Ss configuration table allows configuration of Central
Frequency, Power and Application codes as described in [ITU.G698.2]
and G.694.1 [ITU.G694.1] and other parameters related to new high
speed coherent interfaces.

Number of subcarriers:
This parameter indicates the number of subcarriers available for
the super-channel in case the Transceiver can support multipla
carrier Circuits.

Current Laser Output power:
This parameter report the current Transceiver Output power, it can
be either a setting and measured value (R/W).
Central frequency (see G.694.1 Table 1):
This parameter indicates the Central frequency value that Ss and Rs will be set to work (in THz). See the details in Section 6/ G.694.1 or based on "n" and "k" values in case of multicarrier transceivers (R/W).

Central frequency granularity:
This parameter indicates the Central frequency granularity supported by the transceiver, this value is combined with K and n value to calculate the central frequency on the carrier or sub-carriers (R).

Current Laser Input power:
This parameter report the current Transceiver Input power (G).

Minimum channel spacing:
This is the minimum nominal difference in frequency (in GHz) between two adjacent channels (or carriers) depending on the Transceiver characteristics (R).

Bit rate / Baud rate of optical tributary signals:
Optical tributary signal bit (for NRZ signals) rate or Symbol (for Multiple bit per symbol) rate .

FEC Coding:
This parameter indicate what Forward Error Correction (FEC) code is used at Ss and Rs (R/W) (not mentioned in G.698). .

Maximum bit error ratio (BER):
This parameter indicate the maximum Bit error rate can be supported by the application at the Receiver. In case of FEC applications it is intended after the FEC correction (R) .

Wavelength Range (see G.694.1): [ITU.G694.1]
This parameter indicate minimum and maximum wavelength spectrum (R) in a definite wavelength Band (L, C and S).

Modulatoin format:
This parameter indicates the list of supported Modulation Formats and the provisioned Modulation Format. (R/W).

Inter carrier skew:
This parameter indicates, in case of multi-carrier transceivers the maximum skew between the sub-carriers supported by the transceiver (R).
4.2. Parameters at Ss

The following parameters for the interface at point S are defined in G.698.2 [ITU.G698.2].

Maximum and minimum mean channel output power:
The mean launched power at Ss is the average power (in dBm) of a pseudo-random data sequence coupled into the DWDM link. It is defined as the range (Max and Min) of the parameter (R/W).

Minimum and maximum central frequency:
The central frequency is the nominal single-channel frequency (in THz) on which the digital coded information of the particular optical channel is modulated by use of the NRZ line code. The central frequencies of all channels within an application lie on the frequency grid for the minimum channel spacing of the application given in ITU-T Rec. G.694.1. This parameter gives the maximum and minimum frequency interval the channel must be modulated (R).

Maximum spectral excursion:
This is the maximum acceptable difference between the nominal central frequency (in GHz) of the channel and the minus 15 dB points of the transmitter spectrum furthest from the nominal central frequency measured at point Ss. (R)

Maximum transmitter (residual) dispersion OSNR penalty (B.3/G.959.1) [ITU.G959.1]
Defines a reference receiver that this penalty is measured with. Lowest OSNR at Ss with worst case (residual) dispersion minus the Lowest OSNR at Ss with no dispersion. Lowest OSNR at Ss with no dispersion (R).

Minimum side mode suppression ratio, Minimum channel extinction ratio, Eye mask:
Although are defined in G.698.2 are not supported by this draft (R).

Current Laser Output power:
This parameter report the current Transceiver Output power, it can be either a setting and measured value (R/W) NEED TO DISCUSS ON THIS.

4.3. Interface at point Rs

The following parameters for the interface at point R are defined in G.698.2.
4.3.1. Mandatory parameters

Maximum and minimum mean input power:
The maximum and minimum values of the average received power (in dBm) at point Rs. (R)

Minimum optical signal-to-noise ratio (OSNR):
The minimum optical signal-to-noise ratio (OSNR) is the minimum value of the ratio of the signal power in the wanted channel to the highest noise power density in the range of the central frequency plus and minus the maximum spectral excursion (R)

Receiver OSNR tolerance:
The receiver OSNR tolerance is defined as the minimum value of OSNR at point Rs that can be tolerated while maintaining the maximum BER of the application. (R)

Maximum reflectance at receiver:
Although is defined in G.698.2, this parameter is not supported by this draft (R).

4.3.2. Optional parameters

Current Chromatic Dispersion (CD):
Residual Chromatic Dispersion measured at Rx Transceiver port (R).

Current Optical Signal to Noise Ratio (OSNR):
Current Optical Signal to Noise Ratio (OSNR) estimated at Rx Transceiver port (R).

Current Quality factor (Q):
"Q" factor estimated at Rx Transceiver port (R).

4.3.3. Optical path from point Ss to Rs

The following parameters for the optical path from point S and R are defined in G.698.2 and are covered by draft-ggalimbe-ccamp-iv-yang [ITU.G698.2].

4.4. Use Cases

The use cases are described in draft-ietf-ccamp-dwdm-if-mng-ctrl-fwk
4.5. Optical Interface for external transponder in a WDM network

The ietf-ext-xponder-wdm-if is an augment to the ietf-interface. It allows the user to set the application code/vendor transceiver class/ Central frequency and the output power. The module can also be used to get the list of supported application codes/transceiver class and also the Central frequency/output power/input power of the interface.

```
module: ietf-ext-xponder-wdm-if
augment /if:interfaces/if:interface:
  +--rw optIfOChRsSs
    +--rw if-current-application-code
      |   +--rw application-code-id    uint8
      |   +--rw application-code-type  uint8
      |   +--rw application-code-length uint8
      |   +--rw application-code?     string
    +-ro if-supported-application-codes
      +--ro number-application-codes-supported? uint32
      +--ro application-codes-list* [application-code-id]
        |   +--ro application-code-id     uint8
        |   +--rw application-code-type  uint8
        |   +--rw application-code-length uint8
        |   +--ro application-code?     string
      +--rw output-power?                     int32
      +--ro input-power?                      int32
      +--rw central-frequency?                uint32

notifications:
  +---n opt-if-och-central-frequency-change
    |   +--ro if-name?      leafref
    |   +--ro new-central-frequency
    |     +--ro central-frequency? uint32
  +---n opt-if-och-application-code-change
    |   +--ro if-name?      leafref
    |   +--ro new-application-code
    |     +--ro application-code-id? uint8
    |     +--rw application-code-type uint8
    |     +--rw application-code-length uint8
    |     +--ro application-code? string
```

5. Structure of the Yang Module

ietf-ext-xponder-wdm-if is a top level model for the support of this feature.
6. Yang Module

The ietf-ext-xponder-wdm-if is defined as an extension to ietf interfaces.
<CODE BEGINS> file "ietf-ext-xponder-wdm-if.yang"

module ietf-ext-xponder-wdm-if {
  namespace "urn:ietf:params:xml:ns:yang:ietf-ext-xponder-wdm-if";
  prefix ietf-ext-xponder-wdm-if;

  import ietf-interfaces {
    prefix if;
  }

  organization
    "IETF CCAMP"
    "Working Group";

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

    Editor:  Dharini Hiremagalur
    <mailto:dharinih@juniper.net>";

  description
    "This module contains a collection of YANG definitions for
    configuring Optical interfaces.

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

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

  revision "2016-03-17" {
    description
      "Initial revision.";
    reference
      "";
  }

  grouping opt-if-och-application-code {

description "Application code entity."
leaf application-code-id {
  type uint8 {
    range "1..255";
  }
  description "Id for the Application code";
}
leaf application-code-type {
  type uint8 {
    range "0..1";
  }
  description "Type for the Application code
0 - Standard, 1 - Proprietary
When the Type is Proprietary, then the
first 6 octets of the application-code
will be the OUI (organizationally unique
identifier)";
}
leaf application-code-length {
  type uint8 {
    range "1..255";
  }
  description "Number of octets in the Application code";
}
leaf application-code {
  type string {
    length "1..255";
  }
  description "This parameter indicates the
transceiver application code at Ss and Rs as
defined in [ITU.G698.2] Chapter 5.3, that
is/should be used by this interface.
The optIfOChApplicationsCodeList has all the
application codes supported by this
interface.";
}

typedef dbm-t {
  type decimal64 {
    fraction-digits 2;
    range "-50..-30 | -10..5 | 10000000";
  }
}
grouping opt-if-och-application-code-list {
  description "List of Application codes group."
  leaf number-application-codes-supported {
    type uint32;
    description "Number of Application codes supported by this interface";
  }
  list application-code-list {
    key "application-code-id";
    description "List of the application codes";
    uses opt-if-och-application-code;
  }
}

grouping opt-if-och-power {
  description "Interface optical Power";
  leaf output-power {
    type int32;
    units ".01dbm";
    description "The output power for this interface in .01 dBm. The setting of the output power is optional";
  }

  leaf input-power {
    type int32;
    units ".01dbm";
    config false;
    description "The current input power of this interface";
  }
}

grouping channel-ITU {
  description "channel-ITU";
  container channel-t {
    description "wavelength notation according to RFC-6205";
    leaf grid {
      type uint32;
      description "grid type e.g.: 0=reserved, 1=DWDM, 2=CWDM";
    }
    leaf channel-spacing {

grouping channel-flex {
    description "channel-flex";
    container channel-n-m {
        description "Channel N / M Notation to describe the  
MEdiachannel";
        leaf grid {
            type uint32;
            description "grid type e.g.: 0=reserved, 1=DWDM, 2=CWDM";
        }
        leaf channel-spacing {
            type uint32;
            description "DWDM grid e.g.: 1=100GHz, 2=50GHz, 3=25GHz";
        }
        leaf n {
            type uint32;
            description "N Value (Channel n-m notation)";
        }
        leaf m {
            type uint32;
            description "M Value (Channel n-m notation)";
        }
    }
}

grouping feasibility-limit-list {
    list feasibility-limit {
        key "id";
        description "Feasibility limit power / osnr pair";
        leaf id {
            type uint32;
            description "Unique Identifier";
        }
        leaf power {
            type uint32;
            description "Power level for the limit";
        }
    }
}
type decimal64 {
    fraction-digits 2;
}
units "dB";
description "Feasibility power";
}
leaf osnr {
    type decimal64 {
        fraction-digits 2;
    }
description "Feasibility Signal / Noise";
}

description "Ordered list of feasibility limits
(should match order of supported FEC types
given in fec-type-list)."

}

grouping power-failure-low-alarm-grp {
    description "Optical Power failure alarm ";
    leaf power-failure-low {
        type dbm-t;
        units "dBm";
        default -1;
        description "Power Failure Low Value";
    }
}

    grouping opt-if-och-central-frequency {
        description "Interface Central Frequency";
        leaf central-frequency {
            type uint32;
            description "This parameter indicate This parameter indicates the frequency of this interface ";
        }
    }

notification opt-if-och-central-frequency-change {
    description "A change of Central Frequency has been detected.";
    leaf "if-name" {
        type leafref {

path "/if:interfaces/if:interface/if:name";
}
description "Interface name";
}
container new-opt-if-och-central-frequency {
description "The new Central Frequency of the interface";
uses opt-if-och-central-frequency;
}
}
notification opt-if-och-application-code-change {
description "A change of Application code has been detected.";
leaf "if-name" {
type leafref {
  path "/if:interfaces/if:interface/if:name";
}
description "Interface name";
}
container new-application-code {
description "The new application code for the interface";
uses opt-if-och-application-code;
}
}

augment "/if:interfaces/if:interface" {
description "Parameters for an optical interface";
container optIfOChRsSs {
description "RsSs path configuration for an interface";
container if-current-application-code {
description "Current Application code of the interface";
uses opt-if-och-application-code;
}
container if-supported-application-codes {
config false;
description "Supported Application codes of the interface";
uses opt-if-och-application-code-list;
}
uses opt-if-och-power;
}
7. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operation and content.

8. IANA Considerations

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


Registrant Contact: The IESG.

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

This document registers a YANG module in the YANG Module Names registry [RFC6020].

This document registers a YANG module in the YANG Module Names registry [RFC6020].

prefix: ietf-ext-xponder-wdm-if reference: RFC XXXX

9. Acknowledgements

Gert Grammel is partly funded by European Union Seventh Framework Programme under grant agreement 318514 CONTENT.
10. Contributors

Dean Bogdanovic
Juniper Networks
Westford
U.S.A.
email deanb@juniper.net

Bernd Zeuner
Deutsche Telekom
Darmstadt
Germany
email B.Zeuner@telekom.de

Arnold Mattheus
Deutsche Telekom
Darmstadt
Germany
email a.mattheus@telekom.de

Manuel Paul
Deutsche Telekom
Berlin
Germany
email Manuel.Paul@telekom.de

Walid Wakim
Cisco
9501 Technology Blvd
ROSEMONT, ILLINOIS 60018
UNITED STATES
email wwakim@cisco.com

Kam Lam
Nokia
USA
+1 732 331 3476
kam.lam@nokia.com

11. References

11.1. Normative References


11.2. Informative References


Appendix A. Change Log

This optional section should be removed before the internet draft is submitted to the IESG for publication as an RFC.

Note to RFC Editor: please remove this appendix before publication as an RFC.

Appendix B. Open Issues

Note to RFC Editor: please remove this appendix before publication as an RFC.

Authors’ Addresses

A YANG model to manage the optical interface parameters for an external transponder in a WDM network
draft-dharini-ccamp-dwdm-if-param-yang-06

Abstract

This memo defines a Yang model related to the Optical Transceiver parameters characterising coherent 100G and above interfaces. 100G and above Transceivers support coherent modulation, multiple modulation formats, multiple FEC codes including some not yet specified (or by in phase of specification by) ITU-T G.698.2 [ITU.G698.2] or any other ITU-T recommendation. More context about the state of the Coherent transceivers is described in draft-many-coherent-DWDM-if-control. Use cases are described in RFC7698

The Yang model defined in this memo can be used for Optical Parameters monitoring and/or configuration of the endpoints of a multi-vendor IaDI optical link. The use of this model does not guarantee interworking of transceivers over a DWDM. Optical path feasibility and interoperability has to be determined by means outside the scope of this document. The purpose of this model is to program interface parameters to consistently configure the mode of operation of transceivers.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

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/.
### Table of Contents

1. Introduction ................................................. 3
2. The Internet-Standard Management Framework ................. 3
3. Conventions .................................................. 4
4. Overview .................................................... 4
   4.1. Optical Parameters Description ...................... 5
       4.1.1. Parameters at Ss ................................. 5
       4.1.2. Interface at point Rs ......................... 5
   4.2. Use Cases .............................................. 5
   4.3. Optical Interface for external transponder in a WDM network ........................................... 5
5. Structure of the Yang Module ................................ 8
6. Yang Module .................................................. 8
7. Security Considerations ..................................... 21
8. IANA Considerations ........................................ 21
9. Acknowledgements .......................................... 22
10. Contributors ............................................... 22
11. References ................................................ 23
   11.1. Normative References ............................... 23
   11.2. Informative References ............................ 25
Appendix A. Change Log ......................................... 25
Appendix B. Open Issues ....................................... 25
Authors’ Addresses ............................................ 26
1. Introduction

This memo defines a Yang model that translates and obsolete the SNMP mib module defined in draft-galikunze-ccamp-dwdm-if-snmp-mib for managing single channel optical interface parameters of DWDM applications, using the approach specified in G.698.2. This model supports parameters to characterize coherent transceivers found in current implementations to specify the mode of operation. As application identifiers like those specified in ITU-T G.874.1 [ITU.G874.1] are not available we use mode templates instead. A mode template describes transceiver characteristics in detail and can be identified by a mode-id.

This draft refers and supports the RFC7698 and draft-many-coherent-DWDM-if-control.

The YANG model describing and extending the optical parameters allows different vendors and operators to retrieve, provision and exchange information across the multi-vendor IaDI interfaces in an abstract manner.

The they concept introduced by this YANG model is the notion of a mode. A mode is a combination of parameters or parameter ranges that is supported by a transceiver. As an example, operating a device in QPSK mode may use a different FEC and requires less OSNR to reach the FEC limit than the same transceiver operating in QAM16 mode. Given the number of parameters and their possible combinations it is important for vendors to be able to qualify a set of combinations which is the basis to define a mode. The YANG model furthermore provides means to selecting one mode as current-mode from that pre-defined list of modes supported by the transceiver module. Once selected, current-opt-if-och-mode-params provide the means to configure specific parameters at run time and retrieve actual parameters from the module. For example, the frequency is a parameter that can be set within min/max boundaries set by the current mode. Laser Temperature however is a ro parameter available at run-time that can be checked against the mode boundaries and may trigger an event.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

This memo specifies a Yang model for optical interfaces.
3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119]. In the description of OIDs the convention: Set (S) Get (G) and Trap (T) conventions will describe the action allowed by the parameter.

4. Overview

Figure 1 shows a set of reference points, for single-channel connection between transmitters (Tx) and receivers (Rx). Here the DWDM network elements include an OM and an OD (which are used as a pair with the opposing element), one or more optical amplifiers and may also include one or more OADMs.

```
+-------------------------------------------------+
| Ss | DWDM Network Elements | Rs |
+-+-+ | +--+ | +--+ |
| Tx L1--->| \ | +-------- | +-------- | / | --->Rx L1 |
+-+-+ | +--+ | +--+ | |
| Tx L2--->| OM |------| ROADM |------| OD |--->Rx L2 |
+-+-+ | +--+ | +--+ | |
| Tx L3--->| / | DWDM |----| ^ | DWDM |\ |--->Rx L3 |
+-+-+ | +--+ | +--+ | |
| +-------- | +-------- | +-------- | +-------- |
| Rs v | Ss |
+--+ | +--+ |
| RxLx | TxLx |
+--+ | +--+ |

Ss = reference point at the DWDM network element tributary output
Rs = reference point at the DWDM network element tributary input
Lx = Lambda x
OM = Optical Mux
OD = Optical Demux
ROADM = Reconfigurable Optical Add Drop Mux
```

from Fig. 5.1/G.698.2

Figure 1: External transponder in WDM networks
4.1. Optical Parameters Description

The link between the external transponders through a WDM network media channels are managed at the edges, i.e. at the transmitters (Tx) and receivers (Rx) attached to the S and R reference points respectively.

Definitions of the optical parameters are provided below to increase the readability of the document.

4.1.1. Parameters at Ss

output-power:
The mean launched power at Ss is the average power (in dBm) of a pseudo-random data sequence coupled into the DWDM link.

central frequency:
This parameter indicates the Central frequency value that Ss and Rs will be set to work (in THz)

4.1.2. Interface at point Rs

input-power:
The average received power (in dBm) at point Rs.

Curr-OSNR:
Current Optical Signal to Noise Ratio (OSNR) estimated at Rx Transceiver port.

Curr-q-factor:
"Q" factor estimated at Rx Transceiver port.

4.2. Use Cases

The use cases are described in draft-ietf-ccamp-dwdm-if-mng-ctrl-fwk

4.3. Optical Interface for external transponder in a WDM network

The ietf-ext-xponder-wdm-if is an augment to the ietf-interface. It allows the user to set the operating mode of transceivers as well as other operational parameters. The module provides also threshold settings and notifications to supervise measured parameters and notify the client.

module: ietf-ext-xponder-wdm-if
augment /if:interfaces/if:interface:
  +--rw optIfOChRsSs
    +--rw if-current-mode
| ++--ro mode-id? | string |
| ++--ro min-central-frequency? | uint32 |
| ++--ro max-central-frequency? | uint32 |
| ++--ro min-input-power? | dbm-t |
| ++--ro max-input-power? | dbm-t |
| ++--ro min-output-power? | dbm-t |
| ++--ro max-output-power? | dbm-t |
| ++--ro osnr-margin? | int32 |
| ++--ro q-margin? | int32 |
| ++--ro fec-info? | string |
| ++--ro fec-bitrate? | string |
| ++--ro fec-gain? | string |
| ++--rw fec-ber-mantissa-threshold? | uint32 |
| ++--rw fec-ber-exponent-threshold? | int32 |
| ++--ro number-of-lanes? | uint32 |
| ++--ro min-laser-temperature? | int32 |
| ++--ro max-laser-temperature? | int32 |
| ++--ro min-rx-optical-power? | dbm-t |
| ++--ro max-rx-optical-power? | dbm-t |
| ++--ro min-chromatic-dispersion? | int32 |
| ++--ro max-chromatic-dispersion? | int32 |
| ++--ro min-diff-group-delay? | int32 |
| ++--ro max-diff-group-delay? | int32 |
| ++--ro modulation-format? | string |
| ++--rw bits-per-symbol? | uint32 |
| ++--rw num-symbols-in-alphabet? | uint32 |
| ++--rw symbols-index? | uint32 |
| ++--ro i-center? | int32 |
| ++--ro q-center? | int32 |
| ++--ro i-noise-variance? | int32 |
| ++--ro q-noise-variance? | int32 |
| ++--ro a-noise-variance? | int32 |
| ++--ro p-noise-variance? | int32 |
| +--ro if-supported-mode |
| ++--ro number-of-modes-supported? | uint32 |
| ++--ro mode-list* [mode-id] |
| +--ro mode-id | string |
| +--ro min-central-frequency? | uint32 |
| +--ro max-central-frequency? | uint32 |
| +--ro min-input-power? | dbm-t |
| +--ro max-input-power? | dbm-t |
| +--ro min-output-power? | dbm-t |
| +--ro max-output-power? | dbm-t |
| +--ro osnr-margin? | int32 |
| +--ro q-margin? | int32 |
| +--ro fec-info? | string |
| +--ro fec-bitrate? | string |
| +--ro fec-gain? | string |
---ro fec-ber-mantissa-threshold? uint32
---ro fec-ber-exponent-threshold? int32
---ro number-of-lanes? uint32
---ro min-laser-temperature? int32
---ro max-laser-temperature? int32
---ro min-rx-optical-power? dbm-t
---ro max-rx-optical-power? dbm-t
---ro min-chromatic-dispersion? int32
---ro max-chromatic-dispersion? int32
---ro min-diff-group-delay? int32
---ro max-diff-group-delay? int32
---ro modulation-format? string
---ro bits-per-symbol? uint32
---ro num-symbols-in-alphabet? uint32
---ro symbols-index? uint32
---ro i-center? int32
---ro q-center? int32
---ro i-noise-variance? int32
---ro q-noise-variance? int32
---ro a-noise-variance? int32
---ro p-noise-variance? int32
++-rw current-opt-if-och-mode-params
  ++-rw mode-id? string
  ++-ro osnr-margin? int32
  ++-ro q-margin? int32
  ++-rw central-frequency? uint32
  ++-rw output-power? int32
  ++-ro input-power? int32
  ++-rw min-fec-ber-mantissa-threshold? uint32
  ++-rw min-fec-ber-exponent-threshold? int32
  ++-rw max-fec-ber-mantissa-threshold? uint32
  ++-rw max-fec-ber-exponent-threshold? int32
  ++-rw number-of-tcas-supported? uint32
  ++-rw mode-list* [tca-type]
    ++-rw tca-type opt-if-och-tca-types
    ++-rw min-threshold? int32
    ++-rw max-threshold? int32
    ++-ro cur-osnr? int32
    ++-ro cur-q-factor? int32
    ++-ro uncorrected-words? uint64
    ++-ro fec-ber-mantissa? uint32
    ++-ro fec-ber-exponent? int32

notifications:
  +++-n opt-if-och-central-frequency-change
  +++-ro if-name? -> /if:interfaces/interface/name
  +++-ro new-opt-if-och-central-frequency
  +++-ro central-frequency? uint32
5. Structure of the Yang Module

ietf-ext-xponder-wdm-if is a top level model for the support of this feature.

6. Yang Module

The ietf-ext-xponder-wdm-if is defined as an extension to ietf interfaces.

<CODE BEGINS> file "ietf-ext-xponder-wdm-if.yang"

module ietf-ext-xponder-wdm-if {  
  namespace "urn:ietf:params:xml:ns:yang:ietf-ext-xponder-wdm-if";  
  prefix ietf-ext-xponder-wdm-if;  
  
  import ietf-interfaces {  
    prefix if;  
  }  
  
  organization  
    "IETF CCAMP  
    Working Group";  
  
  contact  
    "WG Web:  <http://tools.ietf.org/wg/ccamp/>  
    WG List:  <mailto:ccamp@ietf.org>  
    Editor:  Dharini Hiremagalur  
    <mailto:dharinih@juniper.net>";  
  
  description  
    "This module contains a collection of YANG definitions for configuring Optical interfaces.  
    Copyright (c) 2016 IETF Trust and the persons identified as authors of the code. All rights reserved."
typedef dbm-t {
    type decimal64 {
        fraction-digits 2;
        range "-50..-30 | -10..5 | 10000000";
    }
    description "Amplifier Power in dBm ";
}
typedef opt-if-och-tca-types {
    type enumeration {
        enum max-laser-linewdt {
description "The maximum laser linewidth";
}
enum min-tx-power-tca {
    description "The min tx power tca";
}
enum max-tx-power-tca {
    description "The min tx power tca";
}
enum min-rx-power-tca {
    description "The min tx power tca";
}
enum max-rx-power-tca {
    description "The min tx power tca";
}
enum max-pol-power-diff-tca {
    description "The power diff. between polariz. tca";
}
enum max-pol-skew-diff-tca {
    description "The Skew between the two polariz. tca";
}
enum min-frequency-offset-tca {
    description "Min Frequency offset tca";
}
enum max-frequency-offset-tca {
    description "Max Frequency offset tca";
}
enum min-osnr-tca {
    description "Min OSNR tca";
}
enum max-osnr-tca {
    description "Max OSNR tca";
}
enum min-laser-temperature-tca {
    description "The min tx power tca";
}
enum max-laser-temperature-tca {
    description "Temperature tca";
}
enum min-fec-ber-tca {
    description "Min Pre Fec BER tca";
}
enum max-fec-ber-tca {
    description "Max Pre Fec BER tca";
}
enum min-q-tca {
    description "Min Q tca";
}
enum max-q-tca {

description "Max Q tca";
}
}
description " The different types of TCA’s";
}

grouping opt-if-och-power {
  description "Interface optical Power";
  leaf output-power {
    type int32;
    units ".01dbm";
    description "The output power for this interface in .01 dBm. The setting of the output power is optional";
  }
}
leaf input-power {
  type int32;
  units ".01dbm";
  config false;
  description "The current input power of this interface";
}
}

grouping opt-if-och-tca-thresholds {
  description "Thresholds for TCA’s";
  leaf tca-type {
    type opt-if-och-tca-types;
    description "type of the TCA eg TX Power";
  }
  leaf min-threshold {
    type int32;
    description " A TCA is generated if the variable is less than this value";
  }
  leaf max-threshold {
    type int32;
    description " A TCA is generated if the variable is more than this value";
  }
}

grouping opt-if-och-fec {
  description "Fec info";
leaf fec-info {
  type string {
    length "1..255";
  }
  config false;
  description "Fec Type - eg GFEC";
}
leaf fec-bitrate {
  type string {
    length "1..255";
  }
  config false;
  description "Fec Overhead rate ";
}
leaf fec-gain {
  type string {
    length "1..255";
  }
  config false;
  description "Fec Overhead rate ";
}
leaf fec-ber-mantissa-threshold {
  type uint32;
  description " Mantissa of the FEC BER threshold";
}
leaf fec-ber-exponent-threshold {
  type int32;
  description " Exponent of the FEC BER threshold";
}
}

grouping opt-if-och-central-frequency {
  description "Interface Central Frequency";
  leaf central-frequency {
    type uint32;
    description " This parameter indicates the frequency of this interface ";
  }
}

grouping opt-if-och-constellation {
  description "Optical constellation parameters";
  leaf i-center {
    type int32;
units ".0001";
config false;
description "The In-phase coordinate of the selected
   constellation symbol for this mode";
}
leaf q-center {
   type int32;
   units ".0001";
   config false;
   description "The Quadrature coordinate of the selected
   constellation symbol for this mode";
}
leaf i-noise-variance {
   type int32;
   units ".001";
   config false;
   description "The Variance of the in-phase noise
   component for this mode";
}
leaf q-noise-variance {
   type int32;
   units ".001";
   config false;
   description "The Variance of the quadrature noise
   component for this mode";
}
leaf a-noise-variance {
   type int32;
   units ".001";
   config false;
   description "The Variance of the radial noise
   component for this mode";
}
leaf p-noise-variance {
   type int32;
   units ".001";
   config false;
   description "The Variance of the phase noise
   component for this mode";
}
}
grouping opt-if-och-modulation-params {
   description "Optical modulation parameters for the lane";
   leaf modulation-format {
      type string {
         length "1..255";
      }
   }
}
config false;
description
  "Modulation format for this mode";
}
leaf bits-per-symbol {
  type uint32;
description "This parameter the bits per symbol for this mode."
}
leaf num-symbols-in-alphabet {
  type uint32;
description "This parameter the bits per symbol for this mode."
}
leaf symbols-index {
  type uint32;
description "This parameter is the symbol index this mode."
}
uses opt-if-och-constellation;
}
grouping opt-if-och-lane-param {
description "Optical parameters for the lane";
leaf number-of-lanes {
  type uint32;
  config false;
description "Number of optical lanes of this interface";
}
leaf min-laser-temperature {
  type int32;
  units ".01C";
  config false;
description "Minimum Laser Temperature of this mode for this interface";
}
leaf max-laser-temperature {
  type int32;
  units ".01C";
  config false;
description "Maximum Laser Temperature of this mode for
leaf min-rx-optical-power {
    type dbm-t;
    config false;
    description
        "Minimum rx optical power of this mode for
         this interface";
}
leaf max-rx-optical-power {
    type dbm-t;
    config false;
    description
        "Maximum rx optical power of this mode for
         this interface";
}
leaf min-chromatic-dispersion {
    type int32;
    config false;
    description
        "Minimum chromatic dispersion of this mode
         for this interface";
}
leaf max-chromatic-dispersion {
    type int32;
    config false;
    description
        "Maximum chromatic dispersion of this
         mode for this interface";
}
leaf min-diff-group-delay {
    type int32;
    config false;
    description
        "Minimum Differential group delay of this
         mode for this interface";
}
leaf max-diff-group-delay {
    type int32;
    config false;
    description
        "Maximum Differential group delay of this
         mode for this interface";
}
uses opt-if-och-modulation-params;
grouping opt-if-och-tca-list {
    description "List of TCA’s.";
    leaf number-of-tcas-supported {
        type uint32;
        description "Number of tcas supported by this interface";
    }
    list mode-list {
        key "tca-type";
        description "List of the tcas";
        uses opt-if-och-tca-thresholds;
    }
}

grouping opt-if-och-fec-tca-thresholds {
    description "Pre FEC BER Thresholds for TCA’s";
    leaf min-fec-ber-mantissa-threshold {
        type uint32;
        description "Min Mantissa of the FEC BER threshold";
    }
    leaf min-fec-ber-exponent-threshold {
        type int32;
        description "Min Exponent of the FEC BER threshold";
    }
    leaf max-fec-ber-mantissa-threshold {
        type uint32;
        description "Max Mantissa of the FEC BER threshold";
    }
    leaf max-fec-ber-exponent-threshold {
        type int32;
        description "Max Exponent of the FEC BER threshold";
    }
}

grouping opt-if-och-mode-params {
    description "OCh mode parameters.";
    leaf mode-id {
        type string {
            length "1..255";
        }
        description "Id for the OCh mode template";
    }
}
leaf osnr-margin {
    type int32;
    units "dB";
    config false;
    description " OSNR margin to FEC threshold";
}
leaf q-margin {
    type int32;
    units "dB";
    config false;
    description " Q-Factor margin to FEC threshold";
}
uses opt-if-och-central-frequency;
uses opt-if-och-power;
uses opt-if-och-fec-tca-thresholds;
uses opt-if-och-tca-list;
}

grouping opt-if-och-statistics {
    description "OCh statistics.";
    leaf cur-osnr {
        type int32;
        units "dB";
        config false;
        description " OSNR margin to FEC threshold";
    }
    leaf cur-q-factor {
        type int32;
        units "dB";
        config false;
        description " Q-Factor of the interface";
    }
    leaf uncorrected-words {
        type uint64;
        config false;
        description " Post FEC errored words";
    }
    leaf fec-ber-mantissa {
        type uint32;
        config false;
        description " Pre fec FEC errored words mantissa";
    }
    leaf fec-ber-exponent {
        type int32;
        config false;
        description " Pre fec FEC errored words exponent";
    }
}
grouping opt-if-och-mode {
  description "OCh mode template.";
  leaf mode-id {
    type string {
      length "1..255";
    }
    config false;
    description "Id for the OCh mode template";
  }
  leaf min-central-frequency {
    type uint32;
    config false;
    description "This parameter indicates the minimum frequency for this template ";
  }
  leaf max-central-frequency {
    type uint32;
    config false;
    description "This parameter indicates the minimum frequency for this template ";
  }
  leaf min-input-power {
    type dbm-t;
    config false;
    description "The minimum input power of this interface";
  }
  leaf max-input-power {
    type dbm-t;
    config false;
    description "The maximum input power of this interface";
  }
  leaf min-output-power {
    type dbm-t;
    config false;
    description "The minimum output power of this interface";
  }
  leaf max-output-power {
    type dbm-t;
    config false;
    description "The maximum output power of this interface";
  }
}
leaf osnr-margin {
    type int32;
    units "dB";
    config false;
    description "OSNR margin to FEC threshold";
}
leaf q-margin {
    type int32;
    units "dB";
    config false;
    description "Q-Factor margin to FEC threshold";
}
uses opt-if-och-fec;
uses opt-if-och-lane-param;

grouping opt-if-och-mode-list {
    description "List of Mode list group.";
    leaf number-of-modes-supported {
        type uint32;
        description "Number of modes supported by this interface";
    }
    list mode-list {
        key "mode-id";
        description "List of the modes";
        uses opt-if-och-mode;
    }
}

notification opt-if-och-central-frequency-change {
    description "A change of Central Frequency has been detected.";
    leaf "if-name" {
        type leafref {
            path "/if:interfaces/if:interface/if:name";
        }
        description "Interface name";
    }
    container new-opt-if-och-central-frequency {
        description "The new Central Frequency of the interface";
        uses opt-if-och-central-frequency;
    }
}
notification opt-if-och-mode-change {
  description "A change of Mode Template has been detected.";
  leaf "if-name" {
    type leafref {
      path "/if:interfaces/if:interface/if:name";
    }
    description "Interface name";
  }
  leaf mode-id {
    type string {
      length "1..255";
    }
    description "Id for the OCh mode template";
  }
}

notification opt-if-och-min-tca {
  description "A min output TCA notification.";
  leaf "if-name" {
    type leafref {
      path "/if:interfaces/if:interface/if:name";
    }
    description "Interface name";
  }
  leaf tca-type {
    type opt-if-och-tca-types;
    description "Type of TCA for eg min tx power TCA";
  }
}
augment "/if:interfaces/if:interface" {
  description "Parameters for an optical interface";
  container optIfOChRsSs {
    description "RsSs path configuration for an interface";
    container if-current-mode {
      description "Current mode template of the interface";
      uses opt-if-och-mode;
    }
  }
  container if-supported-mode {
    config false;
    description "Supported mode list of"
7. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operation and content.

8. IANA Considerations

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


Registrant Contact: The IESG.

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

This document registers a YANG module in the YANG Module Names registry [RFC6020].

This document registers a YANG module in the YANG Module Names registry [RFC6020].

prefix: ietf-ext-xponder-wdm-if reference: RFC XXXX
9. Acknowledgements

10. Contributors

Dean Bogdanovic  
Westford  
U.S.A.  
email ivandean@gmail.com

Bernd Zeuner  
Deutsche Telekom  
Darmstadt  
Germany  
email B.Zeuner@telekom.de

Massimiliano Salsi  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA, 94089  
U.S.A.  
+1 408936847  
email msalsi@juniper.net

Arnold Mattheus  
Deutsche Telekom  
Darmstadt  
Germany  
email a.mattheus@telekom.de

Manuel Paul  
Deutsche Telekom  
Berlin  
Germany  
email Manuel.Paul@telekom.de

Walid Wakim  
Cisco  
9501 Technology Blvd  
ROSEMONT, ILLINOIS 60018  
UNITED STATES  
email wwakim@cisco.com

Kam Lam  
Nokia  
USA  
+1 732 331 3476  
kam.lam@nokia.com
11. References

11.1. Normative References


[ITU.G874]

[ITU.G874.1]

[ITU.G959.1]


11.2. Informative References


Appendix A. Change Log

This optional section should be removed before the internet draft is submitted to the IESG for publication as an RFC.

Note to RFC Editor: please remove this appendix before publication as an RFC.

Appendix B. Open Issues

Note to RFC Editor: please remove this appendix before publication as an RFC.
Authors’ Addresses

Gabriele Galimberti (editor)
Cisco
Via Santa Maria Molgora, 48 c
20871 – Vimercate
Italy
Phone: +390392091462
Email: ggalimbe@cisco.com

Ruediger Kunze
Deutsche Telekom
Winterfeldstr. 21-27
10781 Berlin
Germany
Phone: +491702275321
Email: RKunze@telekom.de

Dharini Hiremagalur (editor)
Juniper
1133 Innovation Way
Sunnyvale - 94089 California
USA
Email: dharinih@juniper.net

Gert Grammel (editor)
Juniper
Oskar-Schlemmer Str. 15
80807 Muenchen
Germany
Phone: +49 1725186386
Email: ggrammel@juniper.net
A YANG Model for IP Link and Transport Service Mapping
draft-fu-pce-ip-link-transport-service-mapping-01

Abstract

IP+optical is a cross-layer collaboration technology for unified management of IP and optical networks. Based on framework proposed in [ACTN-FWK][I-D.ietf-teas-actn-framework], this draft presents specific information about the IP+optical solution: hierarchical controllers + disabled GMPLS UNIs. This solution does not involve UNI tunnel objects. Therefore, the mapping between IP links and transport services is key point of this solution. This draft provides a YANG model for the RESTCONF/NETCONF protocol. This YANG module defines NBIs for the IP+optical super controller.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on May 1, 2017.
1. Introduction

1.1. IP+optical solution

IP+optical is a cross-layer collaboration technology for unified management of IP and optical networks. IP+optical adopts the C/S architecture, where the IP network is the client-layer network and the optical network is the server-layer network. The mapping between IP-layer IP links and transport services is the key ability of an IP+optical network. Through the mapping, the services of IP layers and those of transport layers can be associated to implement use cases of IP+optical scenarios.

IP+optical use cases include multi-layer topology visualization, automated network deployment, multi-layer automated service deployment, multi-layer protection and restoration, multi-layer optimization, and multi-layer maintenance window.
Based on framework proposed in [ACTN-FWK][I-D.ietf-teas-actn-framework], this draft presents specific information about the IP+optical solution: hierarchical controllers + disabled GMPLS UNIs. This solution does not involve UNI tunnel objects. Therefore, the mapping between IP links and transport services is key point of this solution.

The IP+optical solution implements cross-layer service provisioning through cross-layer link and association of multi-layer topologies. After service provisioning, this solution is required to present multi-layer service views for users to learn service status. In addition, the association management function needs to be available during fault demarcation and locating and cross-layer protection and restoration. To meet these demands, a service mapping needs to be maintained between IP-layer IP links and optical-layer transport services.

In real-world situations, IP+optical super controllers can be separately deployed or combined with other controllers. For example, in IP+optical single-domain scenarios, an IP+optical super controller
can be combined with an IP domain controller. In IP multi-domain and optical multi-domain scenarios, you can deploy one separate IP super controller and one separate optical super controller. The two super controllers communicate through RESTConf interfaces and use the IP+VNT algorithm to complete E2E cross-layer path calculation. In such multi-domain scenarios, you can also deploy only one IP+optical super controller and use a unified cross-layer algorithm in the controller to complete E2E cross-layer path calculation.

Figure 2: IP+optical single-domain scenarios
Figure 3: IP domain and optical multi-domain scenarios-1
1.2. Unified cross-layer algorithm

In this model, inter-layer path computation is performed by a single PCE of a Unified controller that has topology visibility into all layers. Such a PCE is called a multi-layer PCE. In Figure 2, the network is comprised of two layers. NEs H1, H2, H3, and H4 belong to the higher layer, and NEs H2, H3, L1, and L2 belong to the lower layer. The PCE is a multi-layer PCE that has visibility into both layers. It can perform end-to-end path computation across layers (single PCE path computation). For instance, it can compute an optimal path H1-H2-L1-L2-H3-H4. Of course, more complex cooperation may be required if an optimal end-to-end path is desired.
1.3. IP+VNT algorithm

In this model, there is at least one PCE of controller per layer, and each PCE of controller has topology visibility restricted to its own layer. Some providers may want to keep the layer boundaries due to factors such as organizational and/or service management issues. The choice for multiple PCE computation instead of single PCE computation may also be driven by scalability considerations, as in this mode a PCE only needs to maintain topology information for one layer (resulting in a size reduction for the Traffic Engineering Database (TED)). Figure 3 shows multiple PCE inter-layer computation with inter-PCE communication. There is one PCE in each layer. The PCEs from each layer collaborate to compute an end-to-end path across layers. An IP-PCE of IP-domain controller uses IP topology and VNT topology information to perform path calculation at the higher layer. If a VNT link is selected, the IP-domain controller collaborates with the optical-domain controller for path calculation. The optical-PCE of optical-domain controller then uses cross-layer topology and optical topology information to calculate an underlying VNT path. A simple example of cooperation between the PCEs could be as follows:

- IP controller sends a request to IP-PCE for a path H1-H4 with ip topo and VNT topo.
o IP-PCE selects VNT link as the entry point and exit point to the lower layer.

o IP-PCE of IP controller requests a path both ends of VNT link from Optical-PCE of optical controller.

o Optical-PCE returns H2-L1-L2-H3 to IP-PCE.

o IP-PCE is now able to compute the full path (H1-H2-L1-L2-H3-H4)

---

1.4. IP Link and Transport Service Mapping

The mapping varies with IP link interfaces and changes with system creation, dismantlement, and scheduling changes.
Figure 7: Physical port connection scenario

Figure 8: VLAN port connection scenario
Figure 9: Eth-trunk port connection scenario

2. IP Link and Transport Service Mapping Model - YANG Tree

(preamble)

module: ietf-mapping-ip_link-transport_service
  +--rw mapping-ip-link-transport-service
     +--rw mappings* [mapping-id]
     |   +--rw mapping-id       string
     |   +--rw mapping-name?    string
     +--rw ip-links
     |   +--rw ip-link* [ip-link-name]
     |     +--rw ip-link-name   string
     |     +--rw ip-link-base?  enumeration
     |     +--rw source-node-id? string
     |     +--rw source-if-id?  uint32
     |     +--rw sink-node-id?  string
     |     +--rw sink-if-id?    uint32
     |     +--rw bandwidth?     decimal64
     |     +--rw delay?         decimal64
     |     +--rw srlg?          decimal64
     |     +--rw ip-optical?    protection-type
     +--rw transport-service
     |   +--rw transport-service-id?  uint32
     |   +--rw bandwidth?          decimal64
     |   +--rw delay-limit?        decimal64
     |   +--rw delayvariation-limit? decimal64
     |   +--rw srlg?               decimal64
     |   +--rw ip-optical?         protection-type
     +--rw supporting-tunnel-name? string

(postamble)
3. IP Link and Transport Service Mapping Model – YANG Code

(preamble)

module ietf-mapping-ip_link-transport_service {
    namespace "urn:ietf:params:xml:ns:yang:
        ietf-mapping-ip_link-transport_service";
    prefix "ip-trans-map";

    organization
        "Huawei Technologies";

    contact
        "fupengcheng@huawei.com";

    description
        "The YANG module defines a mapping between ip link
        and transport.";

    revision 2016-10-28 {
        description "Initial revision.";
    }

    /* Features */
    feature ip-link {
        description "ip-link paras";
    }

    feature transport {
        description "transport paras";
    }

    /* Typedefs */
    typedef protection-type {
        type string;
        description
            "ip or optical protection type.";
    }

    /* Groupings */
    grouping ip-link-paras {
        container ip-links {
            list ip-link {
                key ip-link-name;

                leaf ip-link-name {
                    type string;
                    description
                }
            }
        }
    }

"name of an ip link."
}
leaf ip-link-base {
  type enumeration {
    enum "physical" {
      description "physical link."
    }
    enum "vlan-if" {
      description "vlan if"
    }
    enum "eth-trunk" {
      description "eth-trunk"
    }
  }
}
leaf source-node-id {
  type string;
  description "source node id."
}
leaf source-if-id {
  type uint32;
  description "source if id."
}
leaf sink-node-id {
  type string;
  description "sink node id."
}
leaf sink-if-id {
  type uint32;
  description "sink if id."
}
leaf bandwidth {
  type decimal64 {
    fraction-digits 2;
  }
  description "bandwidth."
}
leaf delay {
  type decimal64 {
    fraction-digits 2;

description "delay.";
}
leaf srlg {
  type decimal64 {
    fraction-digits 2;
  }
  description "srlg.";
}
leaf ip-optical {
  type protection-type;
  description "IP_Optial.";
  description "List of ip links";
}
description "Container of ip links";
}
description "This grouping defines ip link parameters";
}

grouping transport-service-paras {
  container transport-service {
    leaf transport-service-id {
      type uint32;
      description "transport service id.";
    }
    leaf bandwidth {
      type decimal64 {
        fraction-digits 2;
      }
      description "bandwidth.";
    }
    leaf delay-limit {
      type decimal64 {
        fraction-digits 2;
      }
      description "delay limit.";
    }
  }
}
leaf delayvariation-limit {
    type decimal64 {
        fraction-digits 2;
    }
    description "delayvariation limit.";
}
leaf srlg {
    type decimal64 {
        fraction-digits 2;
    }
    description "srlg."
}
leaf ip-optical {
    type protection-type;
    description "IP_Optial.";
}
leaf supporting-tunnel-name {
    type string;
    description "supporting tunnel name."
}
}
description "This grouping defines transport service parameters";
/* Main blocks */
container mapping-ip-link-transport-service {
    list mappings {
        key mapping-id;

        leaf mapping-id {
            type string;
            description "key of a mapping."
        }
        leaf mapping-name {
            type string;
            description "name of a mapping"
        }

        uses ip-link-paras;
        uses transport-service-paras;
    }
}
4. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

6. Acknowledgements

7. Normative References

[I-D.ietf-teas-actn-framework]


Authors’ Addresses

Pengcheng FU
Huawei Technologies

Email: fupengcheng@huawei.com

Jianzhong FU
Huawei Technologies

Email: fujianzhong@huawei.com

Aijun.Wang
China Telecom

Email: wangaj@ctbri.com.cn

Fu, et al. Expires May 1, 2017
Abstract

IP+optical is a cross-layer collaboration technology for unified management of IP and optical networks.

Based on framework proposed in [ACTN-FWK][I-D.ietf-teas-actn-framework], this draft presents specific information about the IP+optical solution: hierarchical controllers + disabled GMPLS UNIs. This solution does not involve UNI tunnel objects.

The system uses loose-coupled dual-controllers to implement cross-layer collaborative path calculation on the virtual network topology (VNT), where the VNT provides the E2E cross-pathcalculation bridging function. The VNT needs to be defined as a YANG model for configuration management.

This draft provides a YANG model for the RESTCONF/NETCONF protocol. This YANG module defines NBIs for the IP+optical super controller.

Requirements Language

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

Status of This Memo

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

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

1.1. IP+optical solution

IP+optical is a cross-layer collaboration technology for unified management of IP and optical networks. IP+optical adopts the C/S architecture, where the IP network is the client-layer network and the optical network is the server-layer network.
IP+optical use cases include multi-layer topology visualization, automated network deployment, multi-layer automated service deployment, multi-layer protection and restoration, multi-layer optimization, and multi-layer maintenance window.

Based on framework proposed in [ACTN-FWK][I-D.ietf-teas-actn-framework], this draft presents specific information about the IP+optical solution: hierarchical controllers + disabled GMPLS UNIs. This solution does not involve UNI tunnel objects. Therefore, the mapping between IP links and transport services is key point of this solution.

![Figure 1: IP+optical solution](image)

In real-world situations, IP+optical super controllers can be separately deployed or combined with other controllers. For example, in IP+optical single-domain scenarios, an IP+optical super controller can be combined with an IP domain controller. In IP multi-domain and optical multi-domain scenarios, you can deploy one separate IP super controller and one separate optical super controller. The two super controllers communicate through RESTConf interfaces and use the IP+VNT algorithm to complete E2E cross-layer path calculation. In
such multi-domain scenarios, you can also deploy only one IP+optical super controller and use a unified cross-layer algorithm in the controller to complete E2E cross-layer path calculation.

The system uses loose-coupled dual-controllers to implement cross-layer collaborative path calculation on the virtual network topology (VNT), where the VNT provides the route calculation bridging function. The VNT needs to be defined as a YANG model for configuration management.

This draft provides a YANG model for the RESTCONF/NETCONF protocol. This YANG module defines NBIs for the IP+optical super controller.

1.2. Unified cross-layer algorithm

In this model, inter-layer path computation is performed by a single PCE of a Unified controller that has topology visibility into all layers. Such a PCE is called a multi-layer PCE. In Figure 2, the network is comprised of two layers. NEs H1, H2, H3, and H4 belong to the higher layer, and NEs H2, H3, L1, and L2 belong to the lower layer. The PCE is a multi-layer PCE that has visibility into both layers. It can perform end-to-end path computation across layers (single PCE path computation). For instance, it can compute an optimal path H1-H2-L1-L2-H3-H4. Of course, more complex cooperation may be required if an optimal end-to-end path is desired.

---
| R | R | \ldots | \ldots | R | R |
| H1 | H2 | \ldots | \ldots | H3 | H4 |
| O | O | \ldots | \ldots | | |
| L1 | L2 |

Figure 2: Unified cross-layer algorithm
1.3. IP+VNT algorithm

In this model, there is at least one PCE of controller per layer, and each PCE of controller has topology visibility restricted to its own layer. Some providers may want to keep the layer boundaries due to factors such as organizational and/or service management issues. The choice for multiple PCE computation instead of single PCE computation may also be driven by scalability considerations, as in this mode a PCE only needs to maintain topology information for one layer (resulting in a size reduction for the Traffic Engineering Database (TED)). Figure 3 shows multiple PCE inter-layer computation with inter-PCE communication. There is one PCE in each layer. The PCEs from each layer collaborate to compute an end-to-end path across layers. An IP-PCE of IP-domain controller uses IP topology and VNT topology information to perform path calculation at the higher layer. If a VNT link is selected, the IP-domain controller collaborates with the optical-domain controller for path calculation. The optical-PCE of optical-domain controller then uses cross-layer topology and optical topology information to calculate an underlying VNT path. A simple example of cooperation between the PCEs could be as follows:

- IP controller sends a request to IP-PCE for a path H1-H4 with IP topo and VNT topo.
- IP-PCE selects VNT link as the entry point and exit point to the lower layer.
- IP-PCE of IP controller requests a path both ends of VNT link from Optical-PCE of optical controller.
- Optical-PCE returns H2-L1-L2-H3 to IP-PCE.
- IP-PCE is now able to compute the full path (H1-H2-L1-L2-H3-H4)
1.4. VNT Protect-Group

VNT links support on-demand creation and deletion, and therefore protection can be implemented based on IP links. To implement the protection function, plan and deploy VNT protection groups. IP link switchover can be then implemented if network faults occur or network traffic reaches a threshold.

VNT protection groups support:

- Manual and automatic service switchover and switchback
- 1:1 and N:1 working modes
- Protection of links with the same source but different sinks, protection of links with different sources but the same sink, and protection of links with both different sources and sinks
Figure 4: VNT protection groups
3. VNT (IP Link) Model - YANG Code

(module: ietf-vnt)

module: ietf-vnt
  +--rw vnts
    +--rw vnt* [vnt-id]
      +--rw vnt-id string
      +--rw vnt-name? string
      +--rw src-node-id? string
      +--rw src-interface-type? string
      +--rw src-interface-ip? inet:ipv4-address
      +--rw src-bind-if-name? string
      +--rw sink-node-id? string
      +--rw sink-interface-type? string
      +--rw sink-interface-ip? inet:ipv4-address
      +--rw sink-bind-if-name? string
      +--rw switch-type? uint16
      +--rw vlan-id? vlan
      +--rw latency? uint32
      +--rw max-reservable-bandwidth? decimal64
      +--rw bandwidth? decimal64
      +--rw te-metric? uint32
      +--rw explicit-path-name? string
      +--rw optical-protection-type? string
typedef vlan {
  type uint16 {
    range "0..4095";
  }
  description "VLAN ID";
}

typedef protection-type {
  type string;
  description "ip or optical protection type.";
}

/* Grouping * /
grouping vnt-para {
  list vnt {
    key "vnt-id";
    description "The list of configured interfaces on the device.";

    leaf vnt-id {
      type string;
      description "Id of vnt.";
    }

    leaf vnt-name {
      type string;
      description "Name of vnt.";
    }

    leaf src-node-id {
      type string;
      description "Id of node.";
    }

    leaf src-interface-type {
      type string;
      description "interface type.";
    }
  }
}
leaf src-interface-ip {
  type inet:ipv4-address;
  description
    "Ip of interface."
}

leaf src-bind-if-name {
  type string;
  description
    "source node bind interface name."
}

leaf sink-node-id {
  type string;
  description
    "Id of node."
}

leaf sink-interface-type {
  type string;
  description
    "interface type."
}

leaf sink-interface-ip {
  type inet:ipv4-address;
  description
    "Ip of interface."
}

leaf sink-bind-if-name {
  type string;
  description
    "sink node bind interface name."
}

leaf switch-type {
  type uint16;
  description
    "switch type."
}

leaf vlan-id {
  type vlan;
  description
    "vlan id."
}
leaf latency {
    type uint32;
    description
        "latency.";
}

leaf max-reservable-bandwidth {
    type decimal64 {
        fraction-digits 2;
    }
    description
        "max reservable bandwidth.";
}

leaf bandwidth {
    type decimal64 {
        fraction-digits 2;
    }
    description
        "bandwidth.";
}

leaf te-metric{
    type uint32;
    description
        "te metric.";
}

leaf explicit-path-name {
    type string;
    description
        "explicit path name";
}

leaf optical-protection-type {
    type string;
    description
        "optical protection type.";
}

/* Main blocks */

container vnts {
    description
        "Virtual network topology.";

    uses vnt-para;

4. VNT (IP Link) Protection Group Model - YANG Tree

(postamble)
5. VNT (IP Link) Protection Group Model - YANG Code

(module ietf-vnt-protect-group {
    namespace "urn:ietf:params:xml:ns:yang:ietf-vnt-protect-group";
    prefix "vnt-protect-grp";

    import ietf-vnt {
        prefix vnt;
    }

    organization "Huawei Technologies";

    contact "fupengcheng@huawei.com";

    description "The YANG module defines vnt protect group.";

    revision 2016-10-28 {
        description "Initial revision.";
    }

    /* Main blocks */
    container vnt-protect-groups {
        description "vnt protect groups.";

        list vnt-protect-group {
            key "group-id";
            description "The list of vnt protect groups.";

            leaf group-id {
                type uint32;
                description "Id of vnt protect group.";
            }

            leaf group-name {
                type string;
                description "Name of vnt protect group.";
            }
        }
    }
}
leaf group-type {
    type enumeration {
        enum "1:1" {
            description "1:1 type.";
        }
        enum "n:1" {
            description "n:1 type";
        }
    }
    description "type of vnt protect group."
}

container work-vnt-list {
    uses vnt:vnt-para;
    description "work vnt list."
}

container protect-vnt-list {
    uses vnt:vnt-para;
    description "protect vnt list."
}

leaf is-autoaction {
    type boolean;
    description "if it is autoaction."
}

leaf is-return {
    type boolean;
    description "if it need return."
}

(postamble)
6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

8. Acknowledgements

9. Normative References

[I-D.ietf-teas-actn-framework]


Authors’ Addresses

Pengcheng FU
Huawei Technologies

Email: fupengcheng@huawei.com

Jianzhong FU
Huawei Technologies

Email: fujianzhong@huawei.com

Aijun.Wang
China Telecom

Email: wangaj@ctbri.com.cn

Ruiquan Jing
China Telecom

Email: jingrq@ctbri.com.cn

Abstract

To ensure an efficient data transport, meeting the requirements requested by today’s IP-services the control and management of DWDM interfaces is a precondition for enhanced multilayer networking and for an further automation of network provisioning and operation. This document describes use cases and requirements for the control and management of optical interfaces parameters according to different types of single channel DWDM interfaces. The focus is on automating the network provisioning process irrespective on how it is triggered i.e. by EMS, NMS or GMPLS. This document covers management as well as control plane considerations in different management cases of a single channel DWDM interface. The purpose is to identify the necessary information elements and processes to be used by control or management systems for further processing.

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 4, 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.  Requirements Language ..................................... 4
2.  Terminology and Definitions ................................... 4
3.  Solution Space Client DWDM interface ............................ 5
3.1.  Comparison of approaches for transverse compatibility ..... 6
3.1.1.  Multivendor DWDM line system with transponders ....... 6
3.1.2.  Integrated single channel DWDM deployments on the client site .................................. 8
4.  Solutions for managing and controlling the optical interface 9
4.1.  Separate Operation and Management Approaches ............... 10
4.1.1.  Direct connection to the management system ............ 10
4.1.2.  Direct connection to the DWDM management system ..... 12
4.2.  Control Plane Considerations .................................. 14
4.2.1.  Considerations using GMPLS UNI ............................ 15
5.  Use cases .................................................... 16
5.1.  Service Setup ................................................ 16
5.2.  Link monitoring Use Cases .................................... 17
5.2.1.  Pure Access Link (AL) Monitoring Use Case ............. 19
5.2.2.  Power Control Loop Use Case ............................... 22
6.  Requirements ................................................ 24
7.  Acknowledgements .............................................. 25
8.  IANA Considerations ............................................ 25
9.  Security Considerations ........................................ 25
10. Contributors .................................................. 26
11. References .................................................... 26
11.1.  Normative References ........................................ 26
11.2.  Informative References ...................................... 28
Authors’ Addresses .................................................. 28
1. Introduction

The usage of the single channel DWDM interfaces in client nodes (e.g. routers) connected to a DWDM Network (which include ROADMs and optical amplifiers) adds a further networking option for operators opening to new scenarios and requiring more control/management plane integration.

Carriers deploy their networks today as a combination of transport and packet infrastructures to ensure high availability and flexible data transport. Both network technologies are usually managed by different operational units using different management concepts. This is the status quo in many carrier networks today. In the case of deployments, where the optical transport interface moves into the client device (e.g., router), it is necessary to coordinate the management of the optical interface at the client domain with the optical transport domain. There are different levels of coordination, which are specified in this framework.

The objective of this document is to provide a framework that describes the solution space for the control and management of single channel interfaces and give use cases on how to manage the solutions. In particular, it examines topological elements and related network management measures. From an architectural point of view, the network can be considered as a set of pre-configured/qualified unidirectional, single-fiber, network connections between reference points S and R shown in figure 2. The optical transport network is managed and controlled in order to provide optical connections at the intended centre frequencies and the optical interfaces are managed and controlled to generate signals of the intended centre frequencies and further parameters as specified for example in ITU-T Recommendations G.698.2 and G.798. The management or control plane of the client and DWDM network be aware of the parameters of the interfaces to properly set up the optical link. This knowledge can be used furthermore, to support fast fault detection.

Optical routing and wavelength assignment based on WSON is out of scope although can benefit of the way the optical parameters are exchanged between the Client and the DWDM Network.

Additionally, the wavelength ordering process and the process how to determine the demand for a new wavelength from A to Z is out of scope.

Note that the Control and Management Planes are two separate entities that are handling the same information in different ways. This document covers management as well as control plane considerations in different management cases of single channel DWDM interfaces.
1.1. Requirements Language

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

2. Terminology and Definitions

Current generation WDM networks are single vendor networks where the optical line system and the transponders are tightly integrated. The DWDM interfaces migration from the Transponders to the Client interfaces changes this scenario, by introducing a standardized interface at the level of OCh between the Client DWDM interface and the DWDM network.

Black Link: The Black Link [ITU.G698.2] allows supporting an optical transmitter/receiver pair of a single vendor or from different vendors to provide a single optical channel interface and transport it over an optical network composed of amplifiers, filters, add-drop multiplexers which may be from a different vendor. Therefore the standard defines the ingress and egress parameters for the optical interfaces at the reference points Ss and Rs.

Single Channel DWDM Interface: The single channel interfaces to DWDM systems defined in G.698.2, which currently include the following features: channel frequency spacing: 50 GHz and wider (defined in [ITU-T G.694.1]); bit rate of single channel: Up to 10 Gbit/s. Future revisions are expected to include application codes for bit rates up to 40 Gb/s.

Forward error correction (FEC): FEC is a way of improving the performance of high-capacity optical transmission systems. Employing FEC in optical transmission systems yields system designs that can accept relatively large BER (much more than 10^-12) in the optical transmission line (before decoding).

Administrative domain [G.805]: For the purposes of this Recommendation an administrative domain represents the extent of resources which belong to a single player such as a network operator, a service provider or an end-user. Administrative domains of different players do not overlap amongst themselves.

Intra-domain interface (IaDI) [G.872]: A physical interface within an administrative domain.

Inter-domain interface (IrDI) [G.872]: A physical interface that represents the boundary between two administrative domains.
Management Plane [G.8081]: The management plane performs management functions for the transport plane, the control plane and the system as a whole. It also provides coordination between all the planes. The following management functional areas are performed in the management plane: performance management; fault management; configuration management; accounting management and security management.

Control Plane [G.8081]: The control plane performs neighbour discovery, call control and connection control functions. Through signalling, the control plane sets up and releases connections, and may restore a connection in case of a failure. The control plane also performs other functions in support of call and connection control, such as neighbour discovery and routing information dissemination.

Transponder: A Transponder is a network element that performs O/E/O (Optical/Electrical/Optical) conversion. In this document it is referred only transponders with 3R (rather than 2R or 1R regeneration) as defined in [ITU.G.872].

Client DWDM interface: A Transceiver element that performs E/O (Electrical/Optical) conversion. In this document it is referred as the DWDM side of a transponder as defined in [ITU.G.872].

3. Solution Space Client DWDM interface

The management of optical interfaces using the Black Link approach deals with aspects related to the management of single-channel optical interface parameters of physical point-to-point and ring DWDM applications on single-mode optical fibres.

The solution allows the direct connection of a wide variety of equipments using a DWDM link, for example:

1. A digital cross-connect with multiple optical interfaces, supplied by a different vendor from the line system

2. Multiple optical client devices, each from a different vendor, supplying one channel each

3. A combination of the above

Table 1 provides a list of management tasks regarding the configuration of optical parameters.
<table>
<thead>
<tr>
<th>Task</th>
<th>Domain</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>determination of centre frequency</td>
<td>optical</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>configuration of centre frequency at optical IF</td>
<td>client</td>
<td>NR</td>
<td>NR</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>path computation of wavelength</td>
<td>optical</td>
<td>NR</td>
<td>NR</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>routing of wavelength</td>
<td>optical</td>
<td>NR</td>
<td>NR</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>wavelength setup across optical network</td>
<td>optical</td>
<td>?</td>
<td>?</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>detection of wavelength fault</td>
<td>client</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>fault isolation, identification of root failure</td>
<td>optical</td>
<td>NR</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>repair actions within optical network</td>
<td>optical</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>protection switching of wavelength restoration of wavelength</td>
<td>optical</td>
<td>NR</td>
<td>NR</td>
<td>R</td>
<td>R</td>
</tr>
</tbody>
</table>

- **Note**: R = relevant, NR = not relevant

**Table 1**: List of tasks related to Client - Network interconnection management

Furthermore the following deployment cases will be considered:

- **a.** Passive WDM
- **b.** P2P WDM systems
- **c.** WDM systems with OADMs
- **d.** Transparent optical networks supporting specific functions, interfaces, protocols etc.

Case a) is added for illustration only, since passive WDM is specified in ITU-T Recommendations G.695 and G.698.1.

Case b) and case c) are motivated by the usage of legacy equipment using the traditional connection as described in Figure 1 DWDM interface integration on the client side.

3.1. Comparison of approaches for transverse compatibility

3.1.1. Multivendor DWDM line system with transponders

As illustrated in Figure 1, for this approach interoperability is achieved via the use of optical transponders providing OEO (allowing conversion to appropriate parameters). The optical interfaces can
then be any short reach standardized optical interface that both vendors support, such as those found in [ITU-T G.957] [ITU-T G.691], [ITU-T G.693], [ITU-T G.959.1], etc.

![Diagram of Inter and Intra-Domain Interface Identification](image)

**TX/RX** = Single channel non-DWDM interfaces  
**T/** = Transponder  
**OM** = Optical Mux  
**OD** = Optical Demux

**Figure 1: Inter and Intra-Domain Interface Identification**

In the scenario of Figure 1 the administrative domain is defined by the Interdomain Interface (IrDI). This interface terminates the DWDM domain. The line side is characterized by the IaDI. This interface specifies the internal parameter set of the optical administrative domain. In the case of a client DWDM interface deployment this interface moves into the client device and extends the optical and administrative domain towards the client node. ITU-T G.698.2 for example specifies the parameter set for a certain set of applications.

This document elaborates only the IaDI Interface as shown in Figure 1 as transversely compatible and multi-vendor interface within one administrative domain controlled by the network operator.
3.1.2. Integrated single channel DWDM deployments on the client site

In case of a deployment as shown in Figure 2, through the use of single channel DWDM interfaces, multi-vendor interconnection can also be achieved while removing the need for one short reach transmitter and receiver pair per channel (eliminating the transponders).

Figure 2 shows a set of reference points, for single-channel connection (Ss and Rs) between transmitters (Tx) and receivers (Rx). Here the DWDM network elements include an optical multiplexer (OM) and an optical demultiplexer (OD) (which are used as a pair with the peer element), one or more optical amplifiers and may also include one or more OADMs.

```
+-------------------------------------------------+
<table>
<thead>
<tr>
<th>Ss</th>
<th>DWDM Network Elements</th>
<th>Rs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time Slot</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------</td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L1</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Tx</td>
<td></td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td></td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-----</td>
</tr>
<tr>
<td>-----</td>
<td></td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-----</td>
</tr>
</tbody>
</table>

Ss = Reference point at the DWDM network element tributary output
Rs = Reference point at the DWDM network element tributary input
Lx = Lambda x
OM = Optical Mux
OD = Optical Demux
OADM = Optical Add Drop Mux
```

Linear DWDM network as per ITU-T G.698.2

Figure 2: Linear Black Link
As shown in Figure 2, the administrative domain may consist of several vendor domains. Even in that case a common north bound management interface is required to ensure a consistent management of the entire connection.

The following documents [DWDM-interface-MIB], [YANG], [LMP] define such a protocol—FIX-THE-REFERENCE specific information using SNMP/SMI, Yang models and LMP TLV to support the direct exchange of information between the client and the network control plane.

4. Solutions for managing and controlling the optical interface

Operation and management of WDM systems is traditionally seen as a homogenous group of tasks that could be carried out best when a single management system or an umbrella management system is used. Currently each WDM vendor provides an Element Management System (EMS) that also administers the wavelengths.

Therefore from the operational point of view the following approaches will be considered to manage and operate optical interfaces.

1. Separate operation and management of client device and the transport network whereas the single channel interface of the client belongs to the administrative domain of the transport network and will be managed by the transport group. This results in two different approaches to send information to the management system:

   a. Direct connection from the client to the management system, ensuring a management of the single channel of the optical network (e.g. EMS, NMS)

   b. Indirect connection to the management system of the optical network using a protocol (LMP) between the client device and the directly connected WDM system node to exchange management information with the optical domain

2. Common operation and management of client device including the single channel DWDM part and the Transport network

The first option keeps the status quo in large carrier networks as mentioned above. In that case it must be ensured that the full FCAPS Management (Fault, Configuration, Accounting, Performance and Security) capabilities are supported. This means from the management staff point of view nothing changes. The transceiver/receiver
optical interface will be part of the optical management domain and will be managed from the transport management staff.

The second solution addresses the case where underlying WDM transport network is mainly used to interconnect a homogeneous set of client nodes (e.g. IP routers or digital crossconnects). Since the service creation and restoration could be done by the higher layers (e.g. IP), this may lead to an efficient network operation and a higher level of integration.

4.1. Separate Operation and Management Approaches

4.1.1. Direct connection to the management system
As depicted in Figure 3 (case 1a) one possibility to manage the optical interface within the client domain is a direct connection to the management system of the optical domain. This ensures manageability as usual.

CL = Client Device
/C = Single Channel Optical Interface
OM = Optical Mux
OD = Optical Demux
EMS = Element Management System
MI= Management Interface

Figure 3: Connecting Single Channel optical interfaces to the Transport Management system

The exchange of management information between client device and the management system assumes that some form of a direct management communication link exists between the client device and the DWDM management system (e.g. EMS). This may be an Ethernet Link or a DCN connection (management communication channel MCC).

It must be ensured that the optical network interface can be managed in a standardised way to enable interoperable solutions between

different optical interface vendors and vendors of the optical network management application. RFC 3591 [RFC3591] defines managed objects for the optical interface type but needs further extension to cover the optical parameters required by this framework document. Therefore an extension to this MIB for the optical interface has been drafted in [DWDM-interface-MIB]. SNMP is used to read parameters and get notifications and alarms, netconf and Yang models are needed to easily provision the interface with the right parameter set as described in [YANG]

Note that a software update of the optical interface components of the client nodes must not lead obligatory to an update of the software of the EMS and vice versa.

4.1.2. Direct connection to the DWDM management system
An alternative as shown in Figure 4 can be used in cases where a more integrated relationship between transport node (e.g. OM or OD) and client device is aspired. In that case a combination of control plane features and manual management will be used.

For information exchange between the client node and the direct connected node of the optical transport network LMP as specified in RFC 4209 [RFC4209] should be used. This extension of LMP may be used between a peer node and an adjacent optical network node as depicted in Figure 4.

The LMP based on RFC 4209 does not yet support the transmission of configuration data (information). This functionality must be added to the existing extensions of the protocol. The use of LMP-WDM assumes that some form of a control channel exists between the client...
node and the WDM equipment. This may be a dedicated lambda, an Ethernet Link, or other signalling communication channel (SCC or IPCC).

4.2. Control Plane Considerations

The concept of integrated single channel DWDM interfaces equally applies to management and control plane mechanisms. The general GMPLS control plane for wavelength switched optical networks is work under definition in the scope of WSON. One important aspect of the BL is the fact that it includes the wavelength that is supported by the given link. Thus a BL can logically be considered as a fiber that is transparent only for a single wavelength. In other words, the wavelength becomes a characteristic of the link itself. Nevertheless the procedure to light up the fiber may vary depending on the implementation. Since the implementation is unknown a priori, different sequences to light up a wavelength need to be considered:

1. Interface first, interface tuning: The transmitter is switched on and the link is immediately transparent to its wavelength. This requires the transmitter to carefully tune power and frequency not overload the line system or to create transients.

2. Interface first, OLS tuning: The transmitter is switched on first and can immediately go to the max power allowed since the OLS performs the power tuning. This leads to an intermediate state where the receiver does not receive a valid signal while the transmitter is sending out one. Alarm suppression mechanisms shall be employed to overcome that condition.

3. OLS first, interface tuning: At first the OLS is tuned to be transparent for a given wavelength, then transponders need to be tuned up. Since the OLS in general requires the presence of a wavelength to fine-tune it is internal facilities there may be a period of time where a valid signal is transmitted but the receiver is unable to detect it. This equally need to be covered by alarm suppression mechanisms.

4. OLS first, OLS tuning: The OLS is programmed to be transparent for a given wavelength, then the interfaces need to be switched on and further power tuning takes place. The sequencing of enabling the link needs to be covered as well.

The preferred way to address these in a Control Plane enabled network is neighbour discovery including exchange of link characteristics and link property correlation. The general mechanisms are covered in RFC4209 [LMP-WDM] and RFC 4204[LMP] which provides the necessary protocol framework to exchange those characteristics between client
and black link. LMP-WDM is not intended for exchanging routing or signalling information but covers:

1. Control channel management
2. Link property correlation
3. Link verification
4. Fault management

Extensions to LMP/LMP-WDM covering the code points of the BL definition are needed. Additionally when client and server side are managed by different operational entities, Link state exchange is required to align the management systems.

4.2.1. Considerations using GMPLS UNI

The deployment of single channel optical interfaces is leading to some functional changes related to the control plane models and has therefore some impact on the existing interfaces especially in the case of an overlay model where the edge node requests resources from the core node and the edges node do not participate in the routing protocol instance that runs among the core nodes. RFC 4208 [RFC4208] defines the GMPLS UNI that will be used between edge and core node. In case of integrated interfaces deployment additional functionalities are needed to setup a connection.

It is necessary to differentiate between topology/signalling information and configuration parameters that are needed to setup a wavelength path. RSVP-TE could be used for the signalling and the reservation of the wavelength path. But there are additional information needed before RSVP-TE can start the signalling process. There are three possibilities to proceed:

a. Using RSVP-TE only for the signalling and LMP as described above to exchange information to configure the optical interface within the edge node or

b. RSVP-TE will be used to transport additional information

c. Leaking IGP information instead of exchanging this information needed from the optical network to the edge node (overlay will be transformed to a border-peer model)

Furthermore following issues should be addressed:
a) The Communication between peering edge nodes using an out of band control channel. The two nodes have to exchange their optical capabilities. An extended version of LMP is needed to exchange FEC Modulation scheme, etc. that must be the same. It would be helpful to define some common profiles that will be supported. Only if the profiles match with both interface capabilities it is possible to start signalling.

b) Due to the bidirectional wavelength path that must be setup it is obligatory that the upstream edge node inserts a wavelength value into the path message for the wavelength path towards the upstream node itself. But in the case of an overlay model the client device may not have full information which wavelength must/should be selected and this information must be exchanged between the edge and the core node.

5. Use cases

A Comparison with the traditional operation scenarios provides an insight of similarities and distinctions in operation and management of single channel optical interfaces. The following use cases provide an overview about operation and maintenance processes.

5.1. Service Setup

It is necessary to differentiate between two operational issues for setting up a light path (a DWDM connection is specific in having defined maximum impairments) within an operational network. The first step is the preparation of the connection if no optical signal is applied. Therefore it is necessary to define the path of the connection.

The second step is to setup the connection between the client DWDM interface and the ROADM port. This is done using the NMS of the optical transport network. From the operation point of view the task is similar in a Black Link scenario and in a traditional WDM environment. The Black Link connection is measured by using BER tester which use optical interfaces according to G.698.2. These measurements are carried out in accordance with [ITU-TG.692]. When needed further connections for resilience are brought into service in the same way.

In addition some other parameters like the transmit optical power, the received optical power, the frequency, etc. must be considered.

If the optical interface moves into a client device some of changes from the operational point of view have to be considered. The centre frequency of the Optical Channel was determined by the setup process.
The optical interfaces at both terminals are set to the centre frequency before interconnected with the dedicated ports of the WDM network. Optical monitoring is activated in the WDM network after the terminals are interconnected with the dedicated ports in order to monitor the status of the connection. The monitor functions of the optical interfaces at the terminals are also activated in order to monitor the end to end connection.

Furthermore it should be possible to automate this last step. After connecting the client device towards the first control plane managed transport node a control connection may e.g. be automatically established using LMP to exchange configuration information.

If tunable interfaces are used in the scenario it would be possible to define a series of backup wavelength routes for restoration that could be tested and stored in backup profile. In fault cases this wavelength routes can be used to recover the service.

5.2. Link monitoring Use Cases

The use cases described below are assuming that power monitoring functions are available in the ingress and egress network element of the DWDM network, respectively. By performing link property correlation it would be beneficial to include the current transmit power value at reference point Ss and the current received power value at reference point Rs. For example if the Client transmitter power (OXC1) has a value of 0dBm and the ROADM interface measured power (at OLS1) is -6dBm the fiber patch cord connecting the two nodes may be pinched or the connectors are dirty. More, the interface characteristics can be used by the OLS network Control Plane in order to check the Optical Channels feasibility. Finally the OXC1 transceivers parameters (Application Code) can be shared with OXC2 using the LMP protocol to verify the transceivers compatibility. The actual route selection of a specific wavelength within the allowed set is outside the scope of LMP. In GMPLS, the parameter selection (e.g. central frequency) is performed by RSVP-TE.

G.698.2 defines a single channel optical interface for DWDM systems that allows interconnecting network-external optical transponders across a DWDM network. The optical transponders are considered to be external to the DWDM network. This so-called 'black link' approach illustrated in Figure 5-1 of G.698.2 and a copy of this figure is provided below. The single channel fiber link between the Ss/Rs reference points and the ingress/egress port of the network element on the domain boundary of the DWDM network (DWDM border NE) is called access link in this contribution. Based on the definition in G.698.2 it is considered to be part of the DWDM network. The access link typically is realized as a passive fiber link that has a specific
optical attenuation (insertion loss). As the access link is an integral part of the DWDM network, it is desirable to monitor its attenuation. Therefore, it is useful to detect an increase of the access link attenuation, for example, when the access link fiber has been disconnected and reconnected (maintenance) and a bad patch panel connection (connector) resulted in a significantly higher access link attenuation (loss of signal in the extreme case of an open connector or a fiber cut). In the following section, two use cases are presented and discussed:

1) pure access link monitoring
2) access link monitoring with a power control loop

These use cases require a power monitor as described in G.697 (see section 6.1.2), that is capable to measure the optical power of the incoming or outgoing single channel signal. The use case where a power control loop is in place could even be used to compensate an increased attenuation as long as the optical transmitter can still be operated within its output power range defined by its application code.
Figure 5 Access Link Power Monitoring

\[
P_{\text{in}} = P_{\text{Tx}} - a_{\text{Tx}}
\]

\[
P_{\text{Rx}} = P_{\text{out}} - a_{\text{Rx}}
\]

- For AL-T monitoring: \( P_{\text{Tx}} \) and \( a_{\text{Tx}} \) must be known
- For AL-R monitoring: \( P_{\text{RX}} \) and \( a_{\text{Rx}} \) must be known

An alarm shall be raised if \( P_{\text{in}} \) or \( P_{\text{Rx}} \) drops below a configured threshold \( t \) [dB]:
- \( P_{\text{in}} < P_{\text{Tx}} - a_{\text{Tx}} - t \) (Tx direction)
- \( P_{\text{Rx}} < P_{\text{out}} - a_{\text{Rx}} - t \) (Rx direction)
- \( a_{\text{Tx}} = a_{\text{Rx}} \)

Figure 5: Extended LMP Model

5.2.1. Pure Access Link (AL) Monitoring Use Case
Figure 6 illustrates the access link monitoring use case and the different physical properties involved that are defined below:

- **Ss, Rs**: Single Channel reference points
- **P(Tx)**: current optical output power of transmitter Tx
- **a(Tx)**: access link attenuation in Tx direction (external transponder point of view)
- **P(in)**: measured current optical input power at the input port of border DWDM NE
- **t**: user defined threshold (tolerance)
- **P(out)**: measured current optical output power at the output port of border DWDM NE
- **a(Rx)**: access link attenuation in Rx direction (external transponder point of view)
- **P(Rx)**: current optical input power of receiver Rx

**Description:**
- The access link attenuation in both directions (a(Tx), a(Rx)) is known or can be determined as part of the commissioning process. Typically, both values are the same.
- A threshold value t has been configured by the operator. This should also be done during commissioning.
- A control plane protocol (e.g. this draft) is in place that allows to periodically send the optical power values P(Tx) and P(Rx) to the control plane protocol instance on the DWDM border NE. This is illustrated in Figure 3.
- The DWDM border NE is capable to periodically measure the optical power Pin and Pout as defined in G.697 by power monitoring points depicted as yellow triangles in the figures below.

**Access Link monitoring process:**
- **Tx direction**: the measured optical input power Pin is compared with the expected optical input power P(Tx) - a(Tx). If the measured optical input power P(in) drops below the value (P(Tx) - a(Tx) - t) a low power alarm shall be raised indicating that the access link attenuation has exceeded a(Tx) + t.
- **Rx direction**: the measured optical input power P(Rx) is compared with the expected optical input power P(out) - a(Rx). If the measured optical input power P(Rx) drops below the value (P(out) - a(Rx) - t) a low power alarm shall be raised indicating that the access link attenuation has exceeded a(Rx) + t.
- to avoid toggling errors, the low power alarm threshold shall be lower than the alarm clear threshold.
Figure 6 Use case 1: Access Link monitoring

- For AL-T monitoring: \( P(Tx) \) and \( a(Tx) \) must be known
- For AL-R monitoring: \( P(Rx) \) and \( a(Rx) \) must be known
An alarm shall be raised if \( P(in) \) or \( P(Rx) \) drops below a configured threshold \( t \) [dB]:
- \( P(in) < P(Tx) - a(Tx) - t \) (Tx direction)
- \( P(Rx) < P(out) - a(Rx) - t \) (Rx direction)
- \( a(Tx) = a(Rx) \)

Figure 6: Extended LMP Model
5.2.2. Power Control Loop Use Case

This use case is based on the access link monitoring use case as described above. In addition, the border NE is running a power control application that is capable to control the optical output power of the single channel tributary signal at the output port of the border DWDM NE (towards the external receiver Rx) and the optical output power of the single channel tributary signal at the external transmitter Tx within their known operating range.

The time scale of this control loop is typically relatively slow (e.g. some 10s or minutes) because the access link attenuation is not expected to vary much over time (the attenuation only changes when re-cabling occurs).

From a data plane perspective, this use case does not require additional data plane extensions. It does only require a protocol extension in the control plane (e.g. this LMP draft) that allows the power control application residing in the DWDM border NE to modify the optical output power of the DWDM domain-external transmitter Tx within the range of the currently used application code. Figure 5 below illustrates this use case utilizing the LMP protocol with extensions defined in this draft.
Figure 7: Power control loop

- The Power Control Loops in Transponder and ROADM controls the Variable Optical Attenuators (VOA) to adjust the proper power in base of the ROADM and Receiver caracteristics and the Access Link attenuation.
6. Requirements

Even if network architectures becomes more complex the management and operation as well as the provisioning process should have a higher degree of automation or should be fully automated. Simplifying and automating the entire management and provisioning process of the network in combination with a higher link utilization and faster restoration times will be the major requirements that has been addressed in this section.

Data Plane interoperability as defined for example in [ITU.G698.2] is a precondition to ensure plain solutions and allow the usage of standardized interfaces between network and control/management plane.

The following requirements are focusing on the usage of standardised integrated single channel interfaces but also valid in other environments.

1. To ensure a lean management and provisioning process of single channel interfaces management and control plane of the client and DWDM network must be aware of the parameters of the interfaces and the optical network to properly setup the optical connection.

2. A standardized northbound API (to network management system) based on Netconf must be supported, alternatively SNMP should be supported too.

3. A standardized data model for single channel interfaces must be supported to exchange optical parameters with control/management plane.

4. Netconf should be used also for configuration of the single channel interfaces including the setting of the power.

5. LMP should be extended and used in cases where optical parameters need to be exchanged between peer nodes to correlate link characteristics and adopt the working mode of the single channel interface.

6. Legacy operational models should be supported (parameters must be exchanged with the DWDM transport EMS to manage the configuration and the transmission of alarms and other FCAPS messages.

7. LMP should be used to adjust the output power of the single channel DWDM interface to ensure that the interface works in the right range defined by the application code.
Parameters e.g. PRE-FEC BER should be used to trigger a FRR mechanism on the IP control plane to reroute traffic before the link breaks.

LMP should be used to automate the end to end connection setup of the optical connection.

Power monitoring functions at both ends of the DWDM connection should be implemented to further automate the setup and shutdown process of the optical interfaces.

A standardized procedure to setup an optical connection should be defined and implemented in DWDM and client devices (containing the single channel optical interface). LMP should be used to ensure that the process follows the right order.

Pre-tested and configured backup paths should be stored in so called backup profiles. In fault cases this wavelength routes can be used to recover the service.

LMP should be used to monitor and observe the access link.

Acknowledgements

The authors would like to thank all who supported the work with fruitful discussions and contributions.

IANA Considerations

This memo includes no request to IANA.

Security Considerations

The architecture and solution space in scope of this framework imposes no additional requirements to the security models already defined in RFC5920 for packet/optical networks using GMPLS, covering also Control Plane and Management interfaces. Respective security mechanisms of the components and protocols, e.g. LMP security models, can be applied unchanged.

As this framework is focusing on the single operator use case, the security concerns can be relaxed to a subset compared to a setup where information is exchanged between external parties and over external interfaces.

Concerning the access control to Management interfaces, security issues can be generally addressed by authentication techniques.
providing origin verification, integrity and confidentiality. Additionally, access to Management interfaces can be physically or logically isolated, by configuring them to be only accessible out-of-band, through a system that is physically or logically separated from the rest of the network infrastructure. In case where management interfaces are accessible in-band at the client device or within the optical transport netork domain, filtering or firewalling techniques can be used to restrict unauthorized in-band traffic. Authentication techniques may be additionally used in all cases.

10. Contributors

Arnold Mattheus
Deutsche Telekom
Darmstadt
Germany
e-mail arnold.Mattheus@telekom.de

Manuel Paul
Deutsche Telekom
Berlin
Germany
e-mail Manuel.Paul@telekom.de

Josef Roese
Deutsche Telekom
Darmstadt
Germany
e-mail j.roese@telekom.de

Frank Luennemann
Deutsche Telekom
Muenster
Germany
e-mail Frank.Luennemann@telekom.de

11. References

11.1. Normative References


11.2. Informative References

[DWDM-interface-MIB]

[ITU-TG.691]

[ITU-TG.693]

[ITU-TG.957]

[ITU-TG.692]

[ITU-TG.959.1]

[ITU-TG.8081]

Authors’ Addresses

A framework for Management and Control of DWDM optical interface parameters
draft-ietf-ccamp-dwdm-if-mng-ctrl-fwk-13

Abstract

The control and management of DWDM interfaces are a precondition for enhanced multilayer networking. They are needed to ensure an efficient data transport, to meet the requirements requested by today’s IP-services and to provide a further automation of network provisioning and operations. This document describes use cases, requirements and solutions for the control and management of optical interface parameters according to different types of single channel DWDM interfaces. The focus is on automating the network provisioning process irrespective on how it is triggered i.e. by Element Manager System (EMS), Network Management System (NMS) or Generalized Multi Protocol Label Switching (GMPLS). This document covers management and control considerations in different scenarios of single channel DWDM interfaces. The purpose is to identify the necessary information and processes to be used by control or management systems to properly and efficiently drive the network.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 25, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction ................................................................. 3
   1.1. Requirements Language ............................................. 4
2. Terminology and Definitions ........................................... 4
3. Solution Space ............................................................ 5
   3.1. Comparison of Approaches for Transverse Compatibility ....... 6
     3.1.1. Multivendor DWDM Line System with Transponders ......... 6
     3.1.2. Integrated Single Channel DWDM Deployments on the Client Site ........................................ 7
4. Solutions for Managing and Controlling Single Channel Optical Interface ........................................... 9
   4.1. Separate Operation and Management Approaches ......... 10
     4.1.1. Direct Connection to the Management System .......... 10
     4.1.2. Indirect Connection to the DWDM Management System (First Optical Node) ........................................ 11
   4.2. Control Plane Considerations ................................. 13
     4.2.1. Considerations Using GMPLS Signaling .................. 14
5. Requirements ............................................................... 15
6. Gap Analysis ............................................................... 16
7. Contributors ............................................................... 18
8. References ................................................................. 18
   8.1. Normative References .............................................. 18
   8.2. Informative References ........................................... 20
Appendix A. Use Cases ......................................................... 21
   A.1. Optical interface parameter collection ....................... 21
   A.2. DWDM client - ROADM interconnection discovery ............. 21
   A.3. Service Setup ........................................................ 21
The usage of external single channel Dense Wavelenght Division Multiplexing (DWDM) interfaces (e.g. in routers) connected to a DWDM Network (e.g. router connected to a network of Reconfigurable Optical Add Drop Multiplexers (ROADM) and optical amplifiers) adds a further networking option for operators but requires an harmonised control and management plane interaction between the different network domains.

Carriers deploy their networks today based on transport and packet network infrastructures as domains to ensure high availability and a high level of redundancy combining the Packet and Transport restoration. Both network domains were operated and managed separately. This is the status quo in many carrier networks today. In the case of deployments where the optical transport interface moves into the client device (e.g. router) an interaction between those domains becomes necessary (e.g. Lambda reprovisioning due to an optical restoration).

This framework specifies different levels of control and management plane interaction to support the usage of single channel optical interfaces in carrier networks in an efficient manner. The interfaces between the two layers can be either gray or coloured.

Although Optical routing and wavelength assignment based on Wavelength Switched Optical Network (WSON) is out of scope, they can benefit from the optical parameters that are exchanged between the Client and the DWDM Network. Also, the wavelength ordering process and determining the demand for a new wavelength path through the network are out of scope. The GMPLS and PCE functions will use the information collected from the Client and the DWDM network, the definition on how PCE and GMPLS can use the information and cooperate to implement RWA and circuit/service provisioning ar aout of scope of this document.

Note that the Control and Management Planes are two separate entities that may handle the same information in different ways.
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, RFC 2119 [RFC2119], RFC 8174 [RFC8174] when, and only when, they appear in all capitals, as shown here.

While RFC 2119 [RFC2119] RFC 8174 [RFC8174] describe interpretations of these key words in terms of protocol specifications and implementations, they are used in this document to describe design requirements for protocol extensions.

2. Terminology and Definitions

The current generation of Wavelength Division Multiplexing (WDM) networks are single vendor networks where the optical line system and the transponders are tightly integrated. The DWDM interface migration from integrated transponders to third party transponders or colored interfaces change this scenario and introduces a standardized interface at the level of OCh between the DWDM interface and the DWDM network.

Black Link: The Black Link [ITU-T.G.698.2] allows supporting an optical transmitter/receiver pair (of a single vendor or from different vendors) to provide a single optical channel interface and transport it over an optical network composed of amplifiers, filters, add-drop multiplexers these being possibly from different vendors. Therefore the standard defines the ingress and egress parameters for the optical interfaces at the reference points Source side (Ss) and Receive side (Rs).

Single Channel DWDM Interface: The single channel interfaces to DWDM systems defined in [ITU-T.G.698.2], which currently include the following features: channel frequency spacing: 50 GHz and wider (defined in [ITU-T.G.694.1] ); bit rate of single channel: Up to 100 Gbit/s. Future revisions are expected to include application codes for bit rates up to 400 Gbit/s.

Forward Error Correction (FEC): FEC is a way of improving the performance of high-capacity optical transmission systems. Employing FEC in optical transmission systems yields system designs that can accept relatively large BER (much more than 10^-12) in the optical transmission line (before decoding).

Administrative domain [ITU-T.G.805]: the extent of resources which belong to a single player such as a network operator, a service provider, or a vendor.
provider or an end-user. Administrative domains of different players do not overlap amongst themselves.

Intra-domain interface (IaDI) [ITU-T.G.872]: A physical interface within an administrative domain.

Inter-domain interface (IrDI) [ITU-T.G.872]: A physical interface that represents the boundary between two administrative domains.

Management Plane [ITU-T.G.8081]: The management plane performs management functions for the transport plane, the control plane and the system as a whole. It also provides coordination between all the planes. The following management functional areas are performed in the management plane: performance management, fault management, configuration management, accounting management and security management.

Control Plane [ITU-T.G.8081]: Through signaling, the control plane sets up and releases connections, may restore a connection in case of a failure, and also performs other functions (e.g., neighbor discovery, topology distribution) in support of those.

Transponder: A Transponder is a network element that performs O/E/O (Optical/Electrical/Optical) conversion. In this document it is referred only transponders with 3R (rather than 2R or 1R regeneration) as defined in [ITU-T.G.872].

Line System: A Line System is a portion of the network including Reconfigurable Add Drop Multiplexers (ROADM) Line Amplifiers and the fibers connecting them.

Client DWDM interface: A Transceiver element that performs E/O (Electrical/Optical) conversion. In this document it is referred as the DWDM side of a transponder as defined in [ITU-T.G.872].

3. Solution Space

The solution space of this document is focusing on aspects related to the management and control of single-channel optical interface parameters of physical point-to-point and ring DWDM applications on single-mode optical fibres and allows the direct connection of a wide variety of equipment using a DWDM link, for example

1. A digital cross-connect with multiple optical interfaces, supplied by a different vendor from the line system

2. Devices as routing, switching or compute nodes, each from a different vendor, providing optical line interfaces
3. A set of Data Center Equipment and servers
4. A combination of the above

3.1. Comparison of Approaches for Transverse Compatibility

This section describes two ways to achieve transverse compatibility. Section 3.1.1 describes the classic model based on well defined inter-domain interfaces. Section 3.1.2 defines a model ensuring interoperability on the line side of the optical network.

3.1.1. Multivendor DWDM Line System with Transponders

As illustrated in Figure 1, for this approach interoperability is achieved via the use of optical transponders providing OEO (allowing conversion to appropriate parameters). The optical interfaces can then be any short reach standardized optical interface that both vendors support, such as those found in [ITU-T.G.957], [ITU-T.G.691], [ITU-T.G.693], etc.

In the scenario of Figure 1 the administrative domain is defined by the Interdomain Interface (IrDI). This interface terminates the DWDM
domain. The line side is characterized by the Intradomain Interface (IaDI). This interface specifies the internal parameter set of the optical administrative domain. In the case of a client DWDM interface deployment this IaDI moves into the client device and extends the optical and administrative domain towards the client node. [ITU-T.G.698.2] for example specifies a set of parameter set for a certain set of applications, see Section 3.1.2.

This document elaborates only the IaDI (Intra Domain Interface) as shown in Figure 1 as DWDM transversely compatible and multi-vendor interface within one administrative domain controlled by the network operator.

SNMP/Simple Management Interface (SMI), NETCONF/RESTCONF and Link Management Protocol (LMP) TLV to support the direct exchange of information between the client and the network management and control plane will be specified in further documents.

The YANG based NETCONF and RESTCONF protocol are better suited for creating and modifying configuration state and thus RECOMMENDED to be used over SNMP MIB. The SNMP MIB creating and modifying configuration state could be used for legacy network.

3.1.2. Integrated Single Channel DWDM Deployments on the Client Site

In case of a deployment as shown in Figure 2, through the use of DWDM interfaces, multi-vendor interconnection can also be achieved. Among the possible use cases, it may be used to remove the need for one short reach transmitter and receiver pair per channel (eliminating the transponders).
Figure 2 shows a set of reference points, for single-channel connection (Ss and Rs) between transmitters (Tx) and receivers (Rx). Here the DWDM network elements include an optical multiplexer (OM) and an optical demultiplexer (OD) (which are used as a pair with the peer element), one or more optical amplifiers and may also include one or more ROADMs.

|=============== Black Link ================|

| Ss | DWDM Network Elements | Rs |
|-----+-----------------------+-----|
| +----+ \        /       +----+ |
| Tx L1 --> \  +--------+  /    --> Rx L1 |
| +----+   \            +----+ |
| Tx L2 --> OM  \   | ROADM | DHDM | OD  --> Rx L2 |
| +----+   \            +----+ |
| Tx L3 --> \  +--------+   \  --> Rx L3 |
| +------------------------+   +--------|
| \ | \ | | | | | | | | | |
| RxLx | Txlx | | | | | | | | | |

Ss = Reference point at the DWDM network element tributary output
Rs = Reference point at the DWDM network element tributary input
Lx = Lambda x
OM = Optical Mux
OD = Optical Demux
ROADM = Reconfigurable Optical Add Drop Mux

Linear DWDM network as per ITU-T G.698.2

Figure 2: Linear Black Link

The single administrative domain may consist of several vendor domains. Even in that case a common network management and control is required to ensure a consistent operation and provisioning of the entire connection.
SNMP/SMI, NETCONF/RESTCONF and LMP TLV to support the direct exchange of information between the client and the network management and control plane will be specified in further documents.

4. Solutions for Managing and Controlling Single Channel Optical Interface

Operation and management of WDM systems is traditionally seen as a homogenous group of tasks that could be carried out best when a single management system or a hierarchical management system is used. Currently each WDM vendor provides an Element Management System (EMS) that also provisions the wavelengths. In a multi-vendor line system, such single-vendor EMS requirement is no more effective. New methods of managing and controlling line systems need to be looked at.

Therefore from the operational point of view the following approaches will be considered to manage and operate optical interfaces.

1. Separate operation and management of client device and the transport network whereas the interface of the client belongs to the administrative domain of the transport network and will be managed by the transport group. This results in two different approaches to send information to the management system

   a. Direct connection from the client node to the transport management system, ensuring a management of the DWDM interface of the optical network (e.g. EMS, NMS)

   b. Indirect connection to the management system of the optical network using a protocol (e.g. LMP) between the client device and the directly connected WDM system node to exchange management information with the optical domain

2. Common operation and management of client device including the single channel DWDM part and the Transport network

The first option keeps the status quo in large carrier networks as mentioned above. In that case it must be ensured that the full FCAPS Management (Fault, Configuration, Accounting, Performance and Security) capabilities are supported. This means from the management staff point of view nothing changes. The transceiver/receiver optical interface will be part of the optical management domain and will be managed from the transport management staff.

The second solution addresses the case where underlying WDM transport network is mainly used to interconnect a homogeneous set of client nodes (e.g. IP routers or digital crossconnects). Since the service creation and restoration could be done by the higher layers (e.g.
IP), this may lead to an efficient network operation and a higher level of integration.

4.1. Separate Operation and Management Approaches

4.1.1. Direct Connection to the Management System

As depicted in Figure 3 (case 1a) one possibility to manage the optical interface within the client domain is a direct connection to the management system of the optical domain. This ensures manageability as usual.

Figure 3: Connecting Single Channel optical interfaces to the Transport Management system

CL = Client Device
/C = Single Channel Optical Interface
OM = Optical Mux
OD = Optical Demux
EMS = Element Management System
MI = Management Interface
DCN = Data Control Network
The exchange of management information between client device and the management system assumes that some form of a direct management communication link exists between the client device and the DWDM management system (e.g. EMS). This may be an Ethernet Link or a DCN connection (management communication channel MCC).

It must be ensured that the optical network interface can be managed in a standardized way to enable interoperable solutions between different optical interface vendors and vendors of the optical network management application. [RFC3591] defines managed objects for the optical interface type but needs further extension to cover the optical parameters required by this framework document.

Is to be noted that the CL (client device) and the DWDM network are belonging to the same operator so the DWDM EMS and the Client devices are connected to the same DCN and the communication security considerations are applicable to CL as per DWDM devices.

Note that a software update of the optical interface components of the client nodes must not lead obligatory to an update of the software of the EMS and vice versa.

4.1.2. Indirect Connection to the DWDM Management System (First Optical Node)
An alternative as shown in Figure 4 should be used in cases where a more integrated relationship between transport node (e.g. OM or OD or ROADM) and client device is aspired. In that case a combination of control plane features and manual management will be used.

For information exchange between the client node and the direct connected node of the optical transport network LMP as specified in RFC 4209 [RFC4209] should be used. This extension of LMP may be used between a peer node and an adjacent optical network node as depicted in Figure 4.

At the time of writing this document, LMP does not yet support the transmission of configuration data (information). This functionality is addressed by draft-ietf-ccamp-dwdm-if-lmp extending the RFC 4209 [RFC4209]. The use of LMP assumes that some form of a control...
channel exists between the client node and the WDM equipment. This may be a dedicated lambda or an Ethernet Link.

4.2. Control Plane Considerations

The concept of integrated single channel DWDM interfaces equally applies to management and control plane mechanisms. GMPLS control plane protocols have been extended for WSON, e.g. RFC 7689 [RFC7689] for fixed grid signal and for flexi-grid RFC 7792 [RFC7792]. One important aspect of the Black Link [ITU-T.G.698.2] is the fact that it is specific to the wavelength that is supported by the given link. Therefore, the link can logically be considered as a fiber that is transparent only for a single wavelength. In other words, the wavelength becomes a characteristic of the link itself.

Nevertheless the procedure to light up the fiber may vary depending on the implementation. Since the implementation is unknown a priori, different sequences to light up a wavelength need to be considered:

1. Interface first, interface tuning: The transmitter is switched on and the link is immediately transparent to its wavelength. This requires the transmitter to carefully tune power and frequency not overload the line system or to create transients.

2. Interface first, Optical Line System (OLS) tuning: The transmitter is switched on first and can immediately go to the max power allowed since the OLS performs the power tuning. This leads to an intermediate state where the receiver does not receive a valid signal while the transmitter is sending out one. Alarm suppression mechanisms shall be employed to overcome that condition.

3. OLS first, interface tuning: At first the OLS is tuned to be transparent for a given wavelength, then transponders need to be tuned up. Since the OLS in general requires the presence of a wavelength to fine-tune its internal facilities there may be a period where a valid signal is transmitted but the receiver is unable to detect it. This equally need to be covered by alarm suppression mechanisms.

4. OLS first, OLS tuning: The OLS is programmed to be transparent for a given wavelength, then the interfaces need to be switched on and further power tuning takes place. The sequencing of enabling the link needs to be covered as well.

The preferred way to address these in a Control Plane enabled network is neighbour discovery including exchange of link characteristics and link property correlation. The general mechanisms are covered in RFC.
4209 [RFC4209] and RFC 4204 [RFC4204] which provides the necessary protocol framework to exchange those characteristics between client and Black Link. LMP-WDM is not intended for exchanging routing or signaling information nor to provision the lambda in the transceiver but covers:

1. Control channel management
2. Link property correlation
3. Link verification
4. Fault management

Extensions to LMP covering the parameter sets (e.g. application codes) are needed, see draft-ietf-ccamp-dwdm-if-lmp. Additionally, when client and server side are managed by different operational entities, the link state may be useful to help the management system to do troubleshooting or alarm correlation.

4.2.1. Considerations Using GMPLS Signaling

The deployment of single channel optical interfaces is leading to some functional changes related to the control plane models and has therefore some impact on the existing interfaces especially in the case of a model where the edge node requests resources from the core node and the edge node do not participate in the routing protocol instance that runs among the core nodes. RFC 4208 [RFC4208] defines the GMPLS UNI that can be used between edge and core node. In case of integrated interfaces deployment additional functionalities are needed to setup a connection.

It is necessary to differentiate between topology/signalling information and configuration parameters that are needed to setup a wavelength path. Using RSVP-TE could be used for the signalling and the reservation of the wavelength path. But there are additional information needed before RSVP-TE can start the signalling process. There are three possibilities to proceed:

a. Using RSVP-TE only for the signalling and LMP as described above to exchange information on the configured optical interface within the edge node

b. RSVP-TE (typically with loose ERO) to transport additional information
c. Leaking IGP information instead of exchanging this information needed from the optical network to the edge node (UNI will be transformed to a border-peer model, see RFC 5146 [RFC5146])

Furthermore following issues should be addressed:

a) The transceivers of peering edge nodes must be compatible. For example, it may be required to know about FEC, modulation scheme, The modulation format, the baudrate and many other parameters described in the drafts reported in the Annex. Depending on where the information is available, compatibility check may either happen before signaling, when the signaling reaches the optical network (e.g. at path computation time), or in the tail end node. An extended version of LMP is needed to exchange this information in case a. above, and to RSVP-TE as well in b. It would be helpful to define some common profiles that will be supported (e.g. the "application identifier") to summarize interface capabilities; if both profiles match, signaling can succeed and provisioning be achieved.

b) Due to the bidirectional wavelength path that must be setup, the upstream edge node must include a wavelength value into the RSVP-TE Path message. But in the case of a UNI model the client device may not have full information about which wavelength must/should be selected, whereas this information must be exchanged between the edge and the core node. The special value defined in [Network-Assigned-Upstream-Label] allows the optical network to assign the actual wavelength to be used by the upstream transponder, which is a simple and efficient solution to this issue.

5. Requirements

As network architectures become more complex, management and operations, including the provisioning process, need progress towards automation. Simplifying and automating the entire management as well as the network provisioning process while enabling higher link utilization and faster restoration times are the main targets of this section.

Supporting network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040] is the base for the communication among EMS/NMS, centralized controller and network elements. This implies to spficy the corresponding IETF YANG modules to fully and consistently manage the feature discussed on this document.

Data plane interoperability as defined for example in [ITU-T.G.698.2] is a precondition to take full benefit from standardized interfaces between network and control/management plane.
The following requirements are focusing on the usage of DWDM interfaces using IETF technologies. Obviously, a common set of solutions must be consistently supported by both the devices hosting DWDM interfaces and the WDM network (i.e., the WDM line). The solutions addressing the following requirements will be discussed in further documents.

1. A YANG data model MUST define the optical parameters to be exchanged (e.g., power setting) between the network elements and the management plane so as to configure single channel interfaces through NETCONF/RESTCONF.

2. LMP MUST allow to convey the relevant optical parameters between two nodes to correlate neighbor characteristics and identify common capabilities or compatible ranges between the WDM line and single channel interfaces.

3. RSVP-TE MUST support the relevant parameters to be exchanged between the device hosting the DWDM interface and the optical node (e.g., the label value), without preventing the network to remain in charge of the optical path computation.

4. Power monitoring functions at both ends of the DWDM connection MAY be used to further automate the setup and shutdown process of the optical interfaces. LMP SHOULD support a way to carry associated measurement from the client devices to the edges of the WDM network.

5. In fault cases, the network SHOULD be able to recover wavelengths. RSVP-TE extensions MUST remain compatible with [RFC4873] features. The Yang modules should mimic a similar level of capability.

6. Gap Analysis

To enable a centralized control function, several gaps in existing RFCs have been identified:
RFC 8343 defines a generic YANG model for interface management. However, to control DWDM interfaces, an augmentation needs to be defined which allows to configure DWDM specifics such as wavelength or FEC-type.

RFC 7224 defines iana-if-type YANG modules and needs extension to include DWDM interfaces.

RFC 4204 defines the Link Management Protocol (LMP) to correlate link properties between two adjacent nodes. Extensions are required to cover the use cases described such as the correlation between a Transponder and a ROADM node.

RFC 8454 defines an information model for Abstraction and Control of TE Networks (ACTN). However it does not support impairment aware path selection or computation.

RFC 7823 describes Performance-Based Path Selection for Explicitly Routed Label Switched Paths (LSPs) Using TE Metric Extensions, but does not define Metric extensions suitable for Impairment aware routing in optical transport Networks.

RFC 7471 in turn defines OSPF Traffic Engineering (TE) Metric Extensions covering several use cases but lacks Impairment awareness.

RFC 6163 provides a Framework for GMPLS and Path Computation Element (PCE) Control of Wavelength Switched Optical Networks (WSONs). While it describes methods for communicating RWA relevant information, it does not identify such information.

Yang Models describing the optical parameter to be used to control the network ad allow an external controller (like ACTN) to are missing although are defined by ITU and reported in the

As this framework is focusing on the single operator use case, the security concerns can be relaxed to a subset compared to a setup where information is exchanged between external parties and over external interfaces.

Concerning the access control to Management interfaces, security issues can be generally addressed by authentication techniques providing origin verification, integrity and confidentiality. Additionally, access to Management interfaces can be physically or logically isolated, by configuring them to be only accessible out-of-band, through a system that is physically or logically separated from the rest of the network infrastructure. In case where management
interfaces are accessible in-band at the client device or within the optical transport network domain, filtering or firewalling techniques can be used to restrict unauthorized in-band traffic. Authentication techniques may be additionally used in all cases.

7. Contributors

Arnold Mattheus  
Deutsche Telekom  
Darmstadt  
Germany  
email arnold.Mattheus@telekom.de

Manuel Paul  
Deutsche Telekom  
Berlin  
Germany  
email Manuel.Paul@telekom.de

Josef Roese  
Deutsche Telekom  
Darmstadt  
Germany  
email j.roese@telekom.de

Frank Luennemann  
Deutsche Telekom  
Muenster  
Germany  
email Frank.Luennemann@telekom.de

Dharini Hiremagalur  
Juniper  
1133 Innovation Way  
Sunnyvale - 94089 California  
USA  
dharinih@juniper.net

8. References

8.1. Normative References

[ITU-T.G.694.1]  


8.2. Informative References


Appendix A. Use Cases

A comparison with the traditional operation scenarios provides an insight of similarities and distinctions in operation and management of DWDM interfaces. The following use cases provide an overview about operation and maintenance processes.

A.1. Optical interface parameter collection

It is necessary to identify the Optical interface characteristics and setting in order to properly calculate the end to end path and match the Head End interface against the Tail End interface compatibility. The optical parameters may have multiple possible values that the Controller (SDN or GMPLS) can use and select for the best network optimisation.

A.2. DWDM client - ROADM interconection discovery

Being the the DWDM port and ROADM port belonging to different domains and Network Elements, the interconnection between them is not embedded in the Optical Nodes and can not be shared to the EMS and the Controller. The Controller needs then to retrieve the connectivity using data coming from the two domains correlating them to discover the relationship. The methods to discover the interconnection can be LMP, LLDP, installation provisioning or any other mechanism checking using the light transmitted by the DWDM transmitter and detector by the ROAMD port photodiode. This use case is fundamental to build the interconnections between the DWDM and Client layer (e.g. Routers) and calculate the multilayer network topology.

A.3. Service Setup

It is necessary to differentiate between different operational issues for setting up a light path (a DWDM connection is specific in having defined maximum impairments) within an operational network.

The first step is to determine if transceivers located at different end-points are interoperable, i.e. support a common set of operational parameters. In this step it is required to determine transceiver capabilities in a way to be able to correlate them for interoperability purposes. Such parameters include modulation scheme, modulation parameters, FEC to name a few. If both transceivers are controlled by the same NMS or CP, such data is readily available. However in cases like Figure 4, a protocol needs to be used to inform the controlling instance (NMS or CP) about transceiver parameters. It is suggested to extend LMP for that purpose.
The second step is to determine the feasibility of a lightpath between two transceivers without applying an optical signal. Understanding the limitations of the transceiver pair, a path through the optical network has to be found, whereby each path has an individual set of impairments deteriorating a wavelength traveling along that path. Since a single transceiver can support multiple parameter sets, the selection of a path may limit the permissible parameter sets determined in previous steps.

The third step is then to setup the connection itself and to determine the Wavelength. This is done using the NMS of the optical transport network or by means of a control plane interaction such as signaling and includes the path information as well as the parameter set information necessary to enable communication.

In a fourth step, optical monitoring is activated in the WDM network in order to monitor the status of the connection. The monitor functions of the optical interfaces at the terminals are also activated in order to monitor the end to end connection.

Furthermore it should be possible to automate this step. After connecting the client device to the neighbor control plane-enabled transport node, a control adjacency may be automatically established, e.g. using LMP.

A.4.  Link Monitoring Use Cases

The use cases described below are assuming that power monitoring functions are available in the ingress and egress network element of the DWDM network, respectively. By performing link property correlation it would be beneficial to include the current transmit power value at reference point Ss and the current received power value at reference point Rs. For example if the Client transmitter power has a value of 0dBm and the ROADM interface measured power is -6dBm the fiber patch cord connecting the two nodes may be pinched or the connectors are dirty. As discussed before, the actual path or selection of a specific wavelength within the allowed set is outside the scope of LMP. The computing entities (e.g. the first optical node originating the circuit) can rely on GMPLS IGP (OSPF) to retrieve all the information related to the network, calculate the path to reach the endpoint and signal the path implementation through the network via RSVP-TE.

[ITU-T.G.698.2] defines a single channel optical interface for DWDM systems that allows interconnecting network-external optical transponders across a DWDM network. The optical transponders are external to the DWDM network. This so-called ‘Black Link’ approach illustrated in Fig. 5-1 of [ITU-T.G.698.2] and a copy of this figure
is provided below in Figure 5. The single channel fiber link between the Ss/Rs reference points and the ingress/egress port of the network element on the domain boundary of the DWDM network (DWDM border NE) is called access link. Based on the definition in [ITU-T.G.698.2] it is part of the DWDM network. The access link is typically realized as a passive fiber link that has a specific optical attenuation (insertion loss). As the access link is an integral part of the DWDM network, it is desirable to monitor its attenuation. Therefore, it is useful to detect an increase of the access link attenuation, for example, when the access link fiber has been disconnected and reconnected (maintenance) and a bad patch panel connection (connector) resulted in a significantly higher access link attenuation (loss of signal in the extreme case of an open connector or a fiber cut). In the following section, two use cases are presented and discussed:

1) pure access link monitoring
2) access link monitoring with a power control loop

These use cases require a power monitor as described in G.697 (see section 6.1.2), that is capable to measure the optical power of the incoming or outgoing single channel signal. The use case where a power control loop is in place could even be used to compensate an increased attenuation if the optical transmitter can still be operated within its output power range defined by its application code.
Use case 1: Access Link monitoring

An alarm shall be raised if P(in) or P(Rx) drops below a configured threshold (t [dB]):

- \( P(\text{in}) < P(\text{Tx}) - a(\text{Tx}) - t \) (Tx direction)
- \( P(\text{Rx}) < P(\text{out}) - a(\text{Rx}) - t \) (Rx direction)
- \( a(\text{Tx}) = a(\text{Rx}) \)

Alarms and events can be shared between Client and Network via LMP.

Figure 5: Access Link Power Monitoring
A.4.1. Pure Access Link (AL) Monitoring Use Case

Figure 6 illustrates the access link monitoring use case and the different physical properties involved that are defined below:

- Ss, Rs: Single Channel reference points
- P(Tx): current optical output power of transmitter Tx
- a(Tx): access link attenuation in Tx direction (external transponder point of view)
- P(in): measured current optical input power at the input port of border DWDM NE
- t: user defined threshold (tolerance)
- P(out): measured current optical output power at the output port of border DWDM NE
- a(Rx): access link attenuation in Rx direction (external transponder point of view)
- P(Rx): current optical input power of receiver Rx

Description:
- The access link attenuation in both directions (a(Tx), a(Rx)) is known or can be determined as part of the commissioning process. Typically, both values are very similar.
- A threshold value t has been configured by the operator. This should also be done during commissioning.
- A control plane protocol is in place that allows to periodically send the optical power values P(Tx) and P(Rx) to the control plane protocol instance on the DWDM border NE. This is illustrated in Figure 3.
- The DWDM border NE is capable to periodically measure the optical power P(in) and P(out) as defined in G.697 by power monitoring points depicted as triangles in the figures below.

Access Link monitoring process:
- Tx direction: the measured optical input power P(in) is compared with the expected optical input power P(Tx) - a(Tx). If the measured optical input power P(in) drops below the value (P(Tx) - a(Tx) - t) a low power alarm shall be raised indicating that the access link attenuation has exceeded a(Tx) + t.
- Rx direction: the measured optical input power P(Rx) is compared with the expected optical input power P(out) - a(Rx). If the measured optical input power P(Rx) drops below the value (P(out) - a(Rx) - t) a low power alarm shall be raised indicating that the access link attenuation has exceeded a(Rx) + t.
- to avoid toggling errors, the low power alarm threshold shall be lower than the alarm clear threshold.
Use case 2: Access Link monitoring through LMP

- For AL-T monitoring: \( P(Tx) \) and \( a(Tx) \) must be known
- For AL-R monitoring: \( P(Rx) \) and \( a(Rx) \) must be known

An alarm shall be raised if \( P(in) \) or \( P(Rx) \) drops below a configured threshold \( t \) [dB]:
- \( P(in) < P(Tx) - a(Tx) - t \) (Tx direction)
- \( P(Rx) < P(out) - a(Rx) - t \) (Rx direction)
- \( a(Tx) = a(Rx) \)

Alarms and events can be shared between Client and Network via LMP according to [RFC4204] and [RFC4209]
A.4.2. Power Control Loop Use Case

This use case is based on the access link monitoring as described above. In addition, the border NE is running a power control application that is capable to control the optical output power of the single channel tributary signal at the output port of the border DWDM NE (towards the external receiver Rx) and the optical output power of the single channel tributary signal at the external transmitter Tx within their known operating range. The time scale of this control loop is typically relatively slow (e.g. some 10s or minutes) because the access link attenuation is not expected to vary much over time (the attenuation only changes when re-cabling occurs).

From a data plane perspective, this use case does not require additional data plane extensions. It does only require a protocol extension in the control plane (e.g. this LMP draft) that allows the power control application residing in the DNDM border NE to modify the optical output power of the DWDM domain-external transmitter Tx within the range of the currently used application code. Figure 7 below illustrates this use case utilizing LMP with the extensions identified in this document.
Use case 3: Power Control Loop

The power control loop in transponders and ROADMs controls the Variable Optical Attenuators (VOA) to adjust the proper power in base of the ROADM and Receiver characteristics and the Access Link attenuation.

Figure 7: Power control loop
A.5. Optical Circuit restoration

Upon a network failure (e.g. fiber cut) the Controller or GMPLS can initiate an Optical Path restoration process. Other than reroute the optical path the controller may need to retune the wavelength and modify the DWDM Transceiver working parameters (e.g. FEC, Modulation Format, etc.). This operation is done in realtime and can benefit of Netconf/Yang interface or RSVP signalling on the UNI interface.

A.6. Multilayer restoration

A network failure can be due to an DWDM port failure. The Controller is the only actor able to fix issue setting a new circuit terminated on a good Client port (GMPLS is not able to make a new path choosing a different end-point). Other than reroute the optical path the controller may need to provision the wavelength and modify the DWDM Transceiver working parameters (e.g. FEC, Modulation Format, etc.). This operation is done in realtime and must be supported by Netconf/Yang interface.

Appendix B. Detailed info drafts

In this section are reported some examples and references on the MIB, Yang and LMP usage. The MIB and TLV defining the parameters described above are reported in the drafts below and are intended as informative data:
draft-ietf-ccamp-dwdm-if-mng-ctrl-fwk-13

Extension to the Link Management Protocol (LMP/DWDM -rfc4209) for Dense Wavelength Division Multiplexing (DWDM) Optical Line Systems to manage the application code of optical interface parameters in DWDM application

draft-ggalimbe-ccamp-flex-if-lmp

Extension to the Link Management Protocol (LMP/DWDM -rfc4209) for Dense Wavelength Division Multiplexing (DWDM) Optical Line Systems to manage the application code of optical interface parameters in DWDM application

draft-ietf-ccamp-dwdm-if-param-yang

A YANG model to manage the optical interface parameters for an external transponder in a WDM network

draft-ietf-ccamp-flexigrid-yang

YANG data model for Flexi-Grid Optical Networks

draft-ietf-ccamp-wson-iv-info

Information Model for Wavelength Switched Optical Networks (WSONs) with Impairments Validation

draft-ietf-ccamp-wson-iv-encode

Information Encoding for WSON with Impairments Validation

draft-galimbe-ccamp-iv-yang

A YANG model to manage the optical parameters for in a WDM network

NOTE: the above information is defined at the time of publication of this document and thus subject to change.

Authors’ Addresses

Ruediger Kunze (editor)
Deutsche Telekom
Winterfeldtstr. 21-27
10781 Berlin
Germany

Phone: +491702275321
Email: RKunze@telekom.de
This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).

Abstract

This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).

Status of This Memo

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

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

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

This Internet-Draft will expire on February 19, 2017.
1.  Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the
network management protocols that defines mechanisms to manage
network devices.  YANG [RFC6020] is a modular language that
represents data structures in an XML tree format, and is used as a
data modelling language for the NETCONF.

This document introduces a YANG data model for MPLS Label
Distribution Protocol (LDP) [RFC5036] and Multipoint LDP (mLDP)
[RFC6388].  For LDP, it also covers LDP IPv6 [RFC7552] and LDP
capabilities [RFC5561].

The data model is defined for following constructs that are used for
managing the protocol:

- Configuration
- Operational State
- Executables (Actions)
- Notifications

This document is organized to define the data model for each of the
above constructs (configuration, state, action, and notifications) in
the sequence as listed earlier.  Given that mLDP is tightly coupled
with LDP, mLDP data model is defined under LDP tree and in the same
sequence as listed above.

2.  Specification of Requirements

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

In this document, the word "IP" is used to refer to both IPv4 and
IPv6, unless otherwise explicitly stated. For example, "IP address
family" means and be read as "IPv4 and/or IPv6 address family"

3.  LDP YANG Model
3.1. Overview

This document defines a new module named "ietf-mpls-ldp" for LDP/mLDP data model where this module augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg].

There are four main containers in "ietf-mpls-ldp" module as follows:

- Read-Write parameters for configuration (Discussed in Section 3.2)
- Read-only parameters for operational state (Discussed in Section 3.3)
- Notifications for events (Discussed in Section 3.4)
- RPCs for executing commands to perform some action (Discussed in Section 3.5)

For the configuration and state data, this model follows the similar approach described in [I-D.openconfig-netmod-opstate] to represent the configuration (intended state) and operational (applied and derived) state. This means that for every configuration (rw) item, there is an associated (ro) item under "state" container to represent the applied state. Furthermore, protocol derived state is also kept under "state" tree corresponding to the protocol area (discovery, peer etc.). [Ed note: This document will be (re-)aligned with [I-D.openconfig-netmod-opstate] once that specification is adopted as a WG document]

Following diagram depicts high level LDP yang tree organization and hierarchy:
Before going into data model details, it is important to take note of the following points:

- This module aims to address only the core LDP/mLDP parameters as per RFC specification, as well as some widely used and deployed non-RFC features (such as label policies, session authentication etc). Any vendor specific feature should be defined in a vendor-specific augmentation of this model.

- Multi-topology LDP [RFC7307] and Multi-topology mLDP [I-D.iwijnand-mpls-mldp-multi-topology] are beyond the scope of this document.

- This module does not cover any applications running on top of LDP and mLDP, nor does it cover any OAM procedures for LDP and mLDP.

- This model is a VPN Forwarding and Routing (VRF)-centric model. It is important to note that [RFC4364] defines VRF tables and default forwarding tables as different, however from a yang modelling perspective this introduces unnecessary complications,
hence we are treating the default forwarding table as just another VRF.

- A "network-instance" as defined in [I-D.rtgyangdt-rtgw-ni-model] refers to a VRF instance (both default and non-default) within the scope of this model.

- This model supports two address-families, namely "ipv4" and "ipv6".

- This model assumes platform-wide label space (i.e. label space Id of zero). However, when Upstream Label assignment [RFC6389] is in use, an upstream assigned label is looked up in a Context-Specific label space as defined in [RFC5331].

- The label and peer policies (including filters) are defined using a prefix-list. When used for a peer policy, the prefix refers to the LSR Id of the peer. The prefix-list is referenced from routing-policy model as defined in [I-D.ietf-rtgwg-policy-model].

- The use of grouping (templates) for bundling and grouping the configuration items is not employed in current revision, and is a subject for consideration in future.

- This model uses the terms LDP "neighbor"/"adjacency", "session", and "peer" with the following semantics:

  * Neighbor/Adjacency: An LDP enabled LSR that is discovered through LDP discovery mechanisms.

  * Session: An LDP neighbor with whom a TCP connection has been established.

  * Peer: An LDP session which has successfully progressed beyond its initialization phase and is either already exchanging the bindings or is ready to do so.

It is to be noted that LDP Graceful Restart mechanisms defined in [RFC3478] allow keeping the exchanged bindings for some time after a session goes down with a peer. We call such a state -- i.e. keeping peer bindings without established or recovered peering -- a "stale" peer. When used in this document, the above terms will refer strictly to the semantics and definitions defined for them.

A graphical representation of LDP YANG data model is presented in Figure 3, Figure 5, Figure 11, and Figure 12. Whereas, the actual model definition in YANG is captured in Section 6.
While presenting the YANG tree view and actual .yang specification, this document assumes the reader is familiar with the concepts of YANG modeling, its presentation and its compilation.

3.2. Configuration

This specification defines the configuration parameters for base LDP as specified in [RFC5036] and LDP IPv6 [RFC7552]. Moreover, it incorporates provisions to enable LDP Capabilities [RFC5561], and defines some of the most significant and commonly used capabilities such as Typed Wildcard FEC [RFC5918], End-of-LIB [RFC5919], and LDP Upstream Label Assignment [RFC6389].

This specification supports VRF-centric configuration. For implementations that support protocol-centric configuration, with provision for inheritance and items that apply to all vrfs, we recommend an augmentation of this model such that any protocol-centric or all-vrf configuration is defined under their designated containers within the standard network-instance (please see Section 3.2.2)

This model augments /rt:routing/rt:control-plane-protocols that is defined in [I-D.ietf-netmod-routing-cfg]. For LDP interfaces, this model refers the MPLS interface as defined under MPLS base specification [I-D.saad-mpls-base-yang]. Furthermore, as mentioned earlier, the configuration tree presents read-write intended configuration leave/items as well as read-only state of the applied configuration. The former is listed under "config" container and latter under "state" container.

Following is high-level configuration organization for LDP/mLDP:
module: ietf-mpls-ldp
  +-- routing
    +-- control-plane-protocols
      +-- mpls-ldp
        +-- global
          +-- ...
          +-- ...
          +-- address-family* [afi]
            +-- . . .
            +-- . . .
        +-- discovery
          +-- . . .
        +-- peers
          +-- ...
          +-- ...

Figure 2

Given the configuration hierarchy, the model allows inheritance such that an item in a child tree is able to derive value from a similar or related item in one of the parent. For instance, hello holdtime can be configured per-VRF or per-VRF-interface, thus allowing inheritance as well flexibility to override with a different value at any child level.

Following is a simplified graphical representation of the data model for LDP configuration:

```yang
+++rw mpls-ldp!
  +++rw global
    +++rw config
      +++rw capability
        +++rw end-of-lib {capability-end-of-lib}?
          |+++rw enable?   boolean
        +++rw typed-wildcard-fec {capability-typed-wildcard-fec}?
          |+++rw enable?   boolean
        +++rw upstream-label-assignment {capability-upstream-label-assign
ment}?
          |+++rw enable?   boolean
        +++rw graceful-restart
          |+++rw enable?   boolean
          |+++rw helper-enable?   boolean {graceful-restart-helper-mod
e}?
          |+++rw reconnect-time?   uint16
          |+++rw recovery-time?   uint16
          |+++rw forwarding-holdtime?   uint16
          |+++rw igp-synchronization-delay?   uint16
          |+++rw lsr-id?   yang:dotted-quad
```
+++rw address-family* [afi]
  +++rw afi       ldp-address-family
  +++rw config
     +++rw enable?         boolean
     +++rw label-policy
        +++rw independent-mode
           +++rw assign {policy-label-assignment-config}?
              +++rw (prefix-option)?
                 |  +++rw prefix-list?        prefix-list-ref
                 |     ++-rw host-routes-only?   boolean
             +++rw advertise
                +++rw explicit-null
                |  +++rw enable?         boolean
                |     ++-rw prefix-list?        prefix-list-ref
             +++rw prefix-list?        prefix-list-ref
             +++rw accept
                +++rw prefix-list?        prefix-list-ref
    +++rw discovery
     +++rw interfaces
        +++rw config
           +++rw hello-holdtime?   uint16
           +++rw hello-interval?   uint16
        +++rw interface* [interface]
           +++rw interface         mpls-interface-ref
          +++rw config
             +++rw hello-holdtime?   uint16
             +++rw hello-interval?   uint16
             +++rw igp-synchronization-delay?   uint16 {per-interface-timer-config}?
        +++rw address-family* [afi]
           +++rw afi       ldp-address-family
           +++rw config
              +++rw enable?         boolean
              +++rw ipv4
                 |  +++rw transport-address?   union
              +++rw ipv6
                 |  +++rw transport-address?   union
           +++rw targeted
```yang
++-rw config
  ++-rw hello-holdtime?    uint16
  ++-rw hello-interval?    uint16
  ++-rw hello-accept {policy-extended-discovery-config}?   
     ++-rw enable?          boolean
     ++-rw neighbor-list?   neighbor-list-ref
++-rw address-family* [afi]
  ++-rw afi     ldp-address-family
++-rw ipv4
  ++-rw target* [adjacent-address]
     ++-rw adjacent-address    inet:ipv4-address
     ++-rw config
     ++-rw enable?          boolean
     ++-rw local-address?   inet:ipv4-address
++-rw ipv6
  ++-rw target* [adjacent-address]
     ++-rw adjacent-address    inet:ipv6-address
     ++-rw config
     ++-rw enable?          boolean
     ++-rw local-address?   inet:ipv6-address
++-rw forwarding-nexthop {forwarding-nexthop-config}?
++-rw interfaces
  ++-rw interface* [interface]
     ++-rw interface         mpls-interface-ref
     ++-rw address-family* [afi]
     ++-rw afi     ldp-address-family
     ++-rw config
     ++-rw ldp-disable?    boolean
++-rw label-policy
  ++-rw independent-mode
     ++-rw assign {policy-label-assignment-config}?
       ++-rw (prefix-option)?   
          ++-rw prefix-list?    prefix-list-ref
          ++-rw host-routes-only?   boolean
     ++-rw advertise
       ++-rw explicit-null
          ++-rw enable?          boolean
          ++-rw prefix-list?    prefix-list-ref
     ++-rw prefix-list?    prefix-list-ref
++-rw ordered-mode {policy-ordered-label-config}?
  ++-rw egress-lsr
     ++-rw prefix-list?    prefix-list-ref
++-rw advertise
     ++-rw prefix-list?    prefix-list-ref
++-rw accept
     ++-rw prefix-list?    prefix-list-ref
```
+++rw peers
  +++rw config
   +++rw session-authentication-md5-password? string
   +++rw session-ka-holdtime? uint16
   +++rw session-ka-interval? uint16
   +++rw session-downstream-on-demand {session-downstream-on-demand-config}?
     +++rw enable? boolean
     +++rw peer-list? peer-list-ref
   +++rw lsr-id yang:dotted-quad
  +++rw config
   +++rw admin-down? boolean
   +++rw capability
   +++rw label-policy
    |  +++rw prefix-list? prefix-list-ref
    +++rw advertise
    |  +++rw prefix-list? prefix-list-ref
    +++rw accept
     |  +++rw prefix-list? prefix-list-ref
   +++rw session-authentication-md5-password? string
   +++rw graceful-restart
    |  +++rw enable? boolean
    |  +++rw reconnect-time? uint16
    |  +++rw recovery-time? uint16
   +++rw session-ka-holdtime? uint16
   +++rw session-ka-interval? uint16
  +++rw address-family
   +++rw ipv4
    |  +++rw label-policy
    |  +++rw advertise
     |  +++rw prefix-list? prefix-list-ref
    |  +++rw accept
     |  +++rw prefix-list? prefix-list-ref
   +++rw ipv6
    |  +++rw label-policy
    |  +++rw advertise
     |  +++rw prefix-list? prefix-list-ref
    |  +++rw accept
     |  +++rw prefix-list? prefix-list-ref

Figure 3

3.2.1. Configuration Hierarchy

The LDP configuration container is logically divided into following high-level config areas:
Per-VRF parameters
  o Global parameters
  o Per-address-family parameters
  o LDP Capabilities parameters
  o Hello Discovery parameters
    - interfaces
      - Per-interface:
        - Global
        - Per-address-family
      - targeted
    - Per-target
  o Peer parameters
    - Global
    - Per-peer
      - Per-address-family
        - Capabilities parameters
  o Forwarding parameters

Figure 4

Following subsections briefly explain these configuration areas.

3.2.1.1. Per-VRF parameters

LDP module resides under an network-instance and the scope of any LDP configuration defined under this tree is per network-instance (per-VRF). This configuration is further divided into sub categories as follows.

3.2.1.1.1. Per-VRF global parameters

There are configuration items that are available directly under a VRF instance and do not fall under any other sub tree. Example of such a parameter is LDP LSR id that is typically configured per VRF. To keep legacy LDP features and applications working in an LDP IPv4 networks with this model, this document recommends an operator to pick a routable IPv4 unicast address as an LSR Id.

3.2.1.1.2. Per-VRF Capabilities parameters

This container falls under global tree and holds the LDP capabilities that are to be enabled for certain features. By default, an LDP capability is disabled unless explicitly enabled. These capabilities are typically used to negotiate with LDP peer(s) the support/non-support related to a feature and its parameters. The scope of a capability enabled under this container applies to all LDP peers in the given VRF instance. There is also a peer level capability
container that is provided to override a capability that is enabled/specified at VRF level.

3.2.1.1.3. Per-VRF Per-Address-Family parameters

Any LDP configuration parameter related to IP address family (AF) whose scope is VRF wide is configured under this tree. The examples of per-AF parameters include enabling LDP for an address family, prefix-list based label policies, and LDP transport address.

3.2.1.1.4. Per-VRF Hello Discovery parameters

This container is used to hold LDP configuration related to Hello and discovery process for both basic (link) and extended (targeted) discovery.

The "interfaces" is a container to configure parameters related to VRF interfaces. There are parameters that apply to all interfaces (such as hello timers), as well as parameters that can be configured per-interface. Hence, an interface list is defined under "interfaces" container. The model defines parameters to configure per-interface non AF related items, as well as per-interface per-AF items. The example of former is interface hello timers, and example of latter is enabling hellos for a given AF under an interface.

The "targeted" container under a VRF instance allows to configure LDP targeted discovery related parameters. Within this container, the "target" list provides a mean to configure multiple target addresses to perform extended discovery to a specific destination target, as well as to fine-tune the per-target parameters.

3.2.1.1.5. Per-VRF Peer parameters

This container is used to hold LDP configuration related to LDP sessions and peers under a VRF instance. This container allows to configure parameters that either apply on VRF’s all peers or a subset (peer-list) of VRF peers. The example of such parameters include authentication password, session KA timers etc. Moreover, the model also allows per-peer parameter tuning by specifying a "peer" list under the "peers" container. A peer is uniquely identified using its LSR Id and hence LSR Id is the key for peer list.

Like per-interface parameters, some per-peer parameters are AF-agnostic (i.e. either non AF related or apply to both IP address families), and some that belong to an AF. The example of former is per-peer session password configuration, whereas the example of latter is prefix-list based label policies (inbound and outbound) that apply to a given peer.
3.2.1.1.6. Per-VRF Forwarding parameters

This container is used to hold configuration used to control LDP forwarding behavior under a VRF instance. One example of a configuration under this container is when a user wishes to enable neighbor discovery on an interface but wishes to disable use of the same interface as forwarding next-hop. This example configuration makes sense only when there are more than one LDP enabled interfaces towards the neighbor.

3.2.2. All-VRFs Configuration

[Ed note: TODO]

3.3. Operational State

Operational state of LDP can be queried and obtained from read-only state containers that fall under the same tree (/rt:routing/rt:control-plane-protocols/) as the configuration.

Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the protocol. [Ed note: This is where this model differs presently from [I-D.openconfig-netmod-opstate] and subject to alignment in later revisions]

Following is a simplified graphical representation of the data model for LDP operational state.

module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
  +-rw mpls-ldp!
     +-rw global
        +-ro state
          +-ro capability
            +-ro end-of-lib {capability-end-of-lib}?
            +-ro typed-wildcard-fec {capability-typed-wildcard-fec}?
            +-ro upstream-label-assignment {capability-upstream-label-assignment}?
          +-ro graceful-restart
            +-ro enable?  boolean
            +-ro helper-enable?  boolean {graceful-restart-helper-module}?
            +-ro reconnect-time?  uint16
            +-ro recovery-time?  uint16
            +-ro forwarding-holdtime?  uint16
++ro  igp-synchronization-delay?  uint16
  +++ro  lsr-id?  yang:dotted-quad
---rw  address-family* [afi]
  +++rw  afi  ldp-address-family
  +++ro  state
    +++ro  enable?  boolean
    +++ro  label-policy
      +++ro  independent-mode
        +++ro  assign {policy-label-assignment-config}?  
          +++ro  (prefix-option)?
            +++:(prefix-list)
              |  +++ro  prefix-list?  prefix-list-ref
              |  +++:(host-routes-only)
                +++ro  host-routes-only?  boolean
        +++ro  advertise
          +++ro  explicit-null
            |  +++ro  enable?  boolean
            |  +++ro  prefix-list?  prefix-list-ref
            +++ro  prefix-list?  prefix-list-ref
          +++ro  accept
            +++ro  prefix-list?  prefix-list-ref
        +++ro  ordered-mode {policy-ordered-label-config}?
          +++ro  egress-lsr
            |  +++ro  prefix-list?  prefix-list-ref
          +++ro  advertise
            +++ro  prefix-list?  prefix-list-ref
          +++ro  accept
            +++ro  prefix-list?  prefix-list-ref
        +++ro  ipv4
          +++ro  transport-address?  inet:ipv4-address
          +++ro  bindings
            +++ro  address* [address]
              |  +++ro  address  inet:ipv4-address
              ++ro  advertisement-type?  advertised-received
              ++ro  peer?  leafref
            ++ro  fec-label* [fec]
              +++ro  fec  inet:ipv4-prefix
              ++ro  peer* [peer advertisement-type]
                |  ++ro  peer  leafref
                ++ro  advertisement-type  advertised-received
                ++ro  label?  mpls:mpls-label
                +++ro  used-in-forwarding?  boolean
            +++ro  ipv6
              +++ro  transport-address?  inet:ipv6-address
              +++ro  binding
                +++ro  address* [address]
                  |  +++ro  address  inet:ipv6-address
                  |  ++ro  advertisement-type?  advertised-received
Internet-Draft      YANG Data Model for LDP and mLDP         August 2016

++-ro peer?       leafref
++-ro fec-label* [fec]
    +++-ro fec    inet:ipv6-prefix
    +++-ro peer* [peer advertisement-type]
        +++-ro peer       leafref
        +++-ro advertisement-type advertised-received
        +++-ro label?     mpls:mpls-label
        +++-ro used-in-forwarding? boolean

++-rw discovery
++-rw interfaces
    +++-ro state
        +++-ro hello-holdtime? uint16
        +++-ro hello-interval? uint16
    ++-rw interface* [interface]
        +++-ro state
            +++-ro hello-holdtime? uint16
            +++-ro hello-interval? uint16
            +++-ro igp-synchronization-delay? uint16 (per-interface-ti
mer-config)?
            | +++-ro next-hello? uint16
        ++-rw address-family* [afi]
            +++-rw afi ldp-address-family
                +++-ro state
                    +++-ro enable? boolean
                    +++-ro ipv4
                        +++-ro transport-address? union
                        +++-ro hello-adjacencies* [adjacent-address]
                            +++-ro adjacent-address inet:ipv4-address
                            +++-ro flag* identityref
                            +++-ro hello-holdtime
                                +++-ro adjacent? uint16
                                +++-ro negotiated? uint16
                                +++-ro remaining? uint16
                            +++-ro next-hello? uint16
                            +++-ro statistics
                                +++-ro discontinuity-time yang:date-and-time
                                +++-ro hello-received? yang:counter64
                                +++-ro hello-dropped? yang:counter64
                                +++-ro peer? leafref
                    +++-ro ipv6
                        +++-ro transport-address? union
                        +++-ro hello-adjacencies* [adjacent-address]
                            +++-ro adjacent-address inet:ipv6-address
                            +++-ro flag* identityref
                            +++-ro hello-holdtime
                                +++-ro adjacent? uint16
                                +++-ro negotiated? uint16
                                +++-ro remaining? uint16
                            +++-ro next-hello? uint16
++-ro statistics
  ++-ro discontinuity-time yang:date-and-time
  ++-ro hello-received? yang:counter64
  ++-ro hello-dropped? yang:counter64
  ++-ro peer? leafref
++-rw targeted
++-ro state
  ++-ro hello-holdtime? uint16
  ++-ro hello-interval? uint16
  ++-ro hello-accept {policy-extended-discovery-config}? boolean
  ++-ro neighbor-list? neighbor-list-ref
++-rw address-family* [afi]
  ++-rw afi ldp-address-family
  ++-ro state
    ++-ro ipv4
      ++-ro hello-adjacencies* [local-address adjacent-address]
      ++-ro local-address inet:ipv4-address
      ++-ro adjacent-address inet:ipv4-address
      ++-ro flag* identityref
      ++-ro hello-holdtime
      ++-ro adjacent? uint16
      ++-ro negotiated? uint16
      ++-ro remaining? uint16
      ++-ro next-hello? uint16
      ++-ro statistics
        ++-ro discontinuity-time yang:date-and-time
        ++-ro hello-received? yang:counter64
        ++-ro hello-dropped? yang:counter64
        ++-ro peer? leafref
    ++-ro ipv6
      ++-ro hello-adjacencies* [local-address adjacent-address]
      ++-ro local-address inet:ipv6-address
      ++-ro adjacent-address inet:ipv6-address
      ++-ro flag* identityref
      ++-ro hello-holdtime
      ++-ro adjacent? uint16
      ++-ro negotiated? uint16
      ++-ro remaining? uint16
      ++-ro next-hello? uint16
      ++-ro statistics
        ++-ro discontinuity-time yang:date-and-time
        ++-ro hello-received? yang:counter64
        ++-ro hello-dropped? yang:counter64
        ++-ro peer? leafref
    ++-rw ipv4
      ++-rw target* [adjacent-address]
      ++-rw adjacent-address inet:ipv4-address
++--rw ipv6
  ++--rw target* [adjacent-address]
    ++--rw adjacent-address inet:ipv6-address
  ++--ro state
    ++--ro enable? boolean
    ++--ro local-address? inet:ipv6-address
++--rw forwarding-nexthop {forwarding-nexthop-config}?
  ++--rw interfaces
    ++--rw interface* [interface]
      ++--rw interface mpls-interface-ref
      ++--rw address-family* [afi]
        ++--rw afi ldp-address-family
      ++--ro state
        ++--ro ldp-disable? boolean
++--rw peers
  ++--ro state
    ++--ro session-authentication-md5-password? string
    ++--ro session-ka-holdtime? uint16
    ++--ro session-ka-interval? uint16
    ++--ro session-downstream-on-demand {session-downstream-on-demand-config}?
      ++--ro enable? boolean
      ++--ro peer-list? peer-list-ref
    ++--rw peer* [lsr-id]
      ++--rw lsr-id yang:dotted-quad
  ++--ro state
    ++--ro admin-down? boolean
    ++--ro capability
      ++--ro label-policy
        ++--ro advertise
          ++--ro prefix-list? prefix-list-ref
        ++--ro accept
          ++--ro prefix-list? prefix-list-ref
      ++--ro session-authentication-md5-password? string
      ++--ro graceful-restart
        ++--ro enable? boolean
        ++--ro reconnect-time? uint16
        ++--ro recovery-time? uint16
    ++--ro session-ka-holdtime? uint16
    ++--ro session-ka-interval? uint16
    ++--ro address-family
      ++--ro ipv4
        ++--ro label-policy
          ++--ro advertise
            ++--ro prefix-list? prefix-list-ref
          ++--ro accept
|  |  |     +--ro prefix-list?   prefix-list-ref
|  |  |     +--ro hello-adjacencies* [local-address adjacent-address]
|  |  |     +--ro local-address       inet:ipv4-address
|  |  |     +--ro adjacent-address    inet:ipv4-address
|  |  |     +--ro flag*               identityref
|  |  |     +--ro hello-holdtime
|  |  |     |  +--ro adjacent?     uint16
|  |  |     |  +--ro negotiated?   uint16
|  |  |     |  +--ro remaining?    uint16
|  |  |     +--ro next-hello?         uint16
|  |  |     +--ro statistics
|  |  |     |  +--ro discontinuity-time    yang:date-and-time
|  |  |     |  +--ro hello-received?       yang:counter64
|  |  |     |  +--ro hello-dropped?        yang:counter64
|  |  |     +--ro interface?          mpls-interface-ref
|  |  |     +--ro ipv6
|  |  |     +--ro label-policy
|  |  |     |  +--ro advertise
|  |  |     |  |  +--ro prefix-list?   prefix-list-ref
|  |  |     |  +--ro accept
|  |  |     |  |  +--ro prefix-list?   prefix-list-ref
|  |  |     +--ro hello-adjacencies* [local-address adjacent-address]
|  |  |     +--ro local-address       inet:ipv6-address
|  |  |     +--ro adjacent-address    inet:ipv6-address
|  |  |     +--ro flag*               identityref
|  |  |     +--ro hello-holdtime
|  |  |     |  +--ro adjacent?     uint16
|  |  |     |  +--ro negotiated?   uint16
|  |  |     |  +--ro remaining?    uint16
|  |  |     +--ro next-hello?         uint16
|  |  |     +--ro statistics
|  |  |     |  +--ro discontinuity-time    yang:date-and-time
|  |  |     |  +--ro hello-received?       yang:counter64
|  |  |     |  +--ro hello-dropped?        yang:counter64
|  |  |     +--ro interface?          mpls-interface-ref
|  |  |     +--ro label-advertisement-mode
|  |  |     +--ro local?        label-adv-mode
|  |  |     +--ro peer?         label-adv-mode
|  |  |     +--ro negotiated?   label-adv-mode
|  |  +--ro graceful-restart
|  |  |  +--ro enable?           boolean
|  |  |  +--ro reconnect-time?   uint16
|  |  |  +--ro recovery-time?    uint16
|  |  +--ro capability
|  |  |  +--ro end-of-lib
|     |  +--ro enable?   boolean
|     +--ro typed-wildcard-fec   
|     |  +--ro enable?   boolean
|     +--ro upstream-label-assignment   
|     |  +--ro enable?   boolean
|     +--ro session-holdtime   
|     |  +--ro peer?         uint16
|     |  +--ro negotiated?   uint16
|     |  +--ro remaining?    uint16
|     +--ro session-state?                         
|     +--ro tcp-connection   
|     |  +--ro local-address?    inet:ip-address
|     |  +--ro local-port?       inet:port-number
|     +--ro remote-address?   inet:ip-address
|     +--ro remote-port?      inet:port-number
|     +--ro up-time?                               
|     +--ro statistics   
|     |  +--ro discontinuity-time          yang:date-and-time
|     |  +--ro received   
|     |     |  +--ro total-octets?          yang:counter64
|     |     |  +--ro total-messages?         yang:counter64
|     |     |  +--ro address?                yang:counter64
|     |     |  +--ro address-withdraw?       yang:counter64
|     |     |  +--ro initialization?         yang:counter64
|     |     |  +--ro keepalive?              yang:counter64
|     |     |  +--ro label-abort-request?    yang:counter64
|     |     |  +--ro label-mapping?          yang:counter64
|     |     |  +--ro label-release?          yang:counter64
|     |     |  +--ro label-request?          yang:counter64
|     |     |  +--ro label-withdraw?         yang:counter64
|     |     |  +--ro notification?           yang:counter64
|     |  +--ro sent   
|     |     |  +--ro total-octets?          yang:counter64
|     |     |  +--ro total-messages?         yang:counter64
|     |     |  +--ro address?                yang:counter64
|     |     |  +--ro address-withdraw?       yang:counter64
|     |     |  +--ro initialization?         yang:counter64
|     |     |  +--ro keepalive?              yang:counter64
|     |     |  +--ro label-abort-request?    yang:counter64
|     |     |  +--ro label-mapping?          yang:counter64
|     |     |  +--ro label-release?          yang:counter64
|     |     |  +--ro label-request?          yang:counter64
|     |     |  +--ro label-withdraw?         yang:counter64
|     |     |  +--ro notification?           yang:counter64
|     |  +--ro total-addresses?            uint32
|     |  +--ro total-labels?               uint32
|     |  +--ro total-fec-label-bindings?   uint32
3.3.1. Derived States

Following are main areas for which LDP operational "derived" state is defined:

- Neighbor Adjacencies
- Peer
- Bindings (FEC-label and address)
- Capabilities

3.3.1.1. Adjacency state

Neighbor adjacencies are per address-family hello adjacencies that are formed with neighbors as result of LDP basic or extended discovery. In terms of organization, there is a source of discovery (e.g. interface or target address) along with its associated parameters and one or more discovered neighbors along with neighbor discovery related parameters. For the basic discovery, there could be more than one discovered neighbor for a given source (interface), whereas there is at most one discovered neighbor for an extended discovery source (local-address and target-address). This is also to be noted that the reason for a targeted neighbor adjacency could be either an active source (locally configured targeted) or passive source (to allow any incoming extended/targeted hellos). A neighbor/adjacency record also contains session-state that helps highlight whether a given adjacency has progressed to subsequent session level or to eventual peer level.

Following captures high level tree hierarchy for neighbor adjacency state.
3.3.1.2. Peer state

Peer related derived state is presented under peers tree. This is one of the core state that provides info on the session related parameters (mode, authentication, KA timeout etc.), TCP connection info, hello adjacencies for the peer, statistics related to messages and bindings, and capabilities exchange info.

Following captures high level tree hierarchy for peer state.
3.3.1.3. Bindings state

Binding state provides information on LDP FEC-label bindings as well as address binding for both inbound (received) as well as outbound (advertised) direction. FEC-label bindings are presented as a FEC-centric view, and address bindings are presented as an address-centric view:
FEC-Label bindings:
FEC 200.1.1.1/32:
    advertised: local-label 16000
    peer 192.168.0.2:0
    peer 192.168.0.3:0
    peer 192.168.0.4:0
    received:
    peer 192.168.0.2:0, label 16002, used-in-forwarding=Yes
    peer 192.168.0.3:0, label 17002, used-in-forwarding=No
FEC 200.1.1.2/32:
    . . .
FEC 201.1.0.0/16:
    . . .

Address bindings:
Addr 1.1.1.1:
    advertised
Addr 1.1.1.2:
    advertised
Addr 2.2.2.2:
    received, peer 192.168.0.2
Addr 2.2.2.22:
    received, peer 192.168.0.2
Addr 3.3.3.3:
    received, peer 192.168.0.3
Addr 3.3.3.33:
    received, peer 192.168.0.3

Figure 8
Note that all local addresses are advertised to all peers and hence no need to provide per-peer information for local address advertisement. Furthermore, note that it is easy to derive a peer-centric view for the bindings from the information already provided in this model.

Following captures high level tree hierarchy for bindings state.
3.3.1.4. Capabilities state

LDP capabilities state comprise two types of information - global information (such as timer etc.), and per-peer information.

Following captures high level tree hierarchy for LDP capabilities state.

```
---rw mpls-ldp!
  +--rw global
    |  +--ro state
    |     +--ro capability
    |     +--ro peer ...
    |     +--ro peer ...
    +--rw peers
      +--rw peer* [lsr-id]
        +--rw lsr-id yang:dotted-quad
        +--ro state
          +--ro received-peer-state
          +--ro capability
          +--ro ....
          +--ro ....
```

Figure 10
3.4. Notifications

This model defines a list of notifications to inform client of important events detected during the protocol operation. These events include events related to changes in the operational state of an LDP peer, hello adjacency, and FEC etc. It is to be noted that an LDP FEC is treated as operational (up) as long as it has at least 1 NHLFE with outgoing label.

Following is a simplified graphical representation of the data model for LDP notifications.

```yang
definitions yang {  module ietf-mpls-ldp {    notifications {      +---n mpls-ldp-peer-event {        +--ro event-type?      oper-status-event-type        +--ro peer-ref?      leafref        +---n mpls-ldp-hello-adjacency-event {          +--ro event-type?      oper-status-event-type          +--ro (hello-adjacency-type)?            +---:targeted {              +--ro targeted {                +--ro target-address?      inet:ip-address              +---:link {                  +--ro link {                    +--ro next-hop-interface?      mpls-interface-ref                    +--ro next-hop-address?      inet:ip-address                  +---n mpls-ldp-fec-event {                    +--ro event-type?      oper-status-event-type                    +--ro prefix?      inet:ip-prefix                 }               }             }           }         }       }   } } }```

Figure 11

3.5. Actions

This model defines a list of rpcs that allow performing an action or executing a command on the protocol. For example, it allows to clear (reset) LDP peers, hello-adjacencies, and statistics. The model makes an effort to provide different level of control so that a user is able to either clear all, or clear all for a given type, or clear a specific entity.

Following is a simplified graphical representation of the data model for LDP actions.
module: ietf-mpls-ldp
rpcs:
  +---x mpls-ldp-clear-peer
    |  +---w input
    |     +---w lsr-id?  union
  +---x mpls-ldp-clear-hello-adjacency
    +---w input
     +---w hello-adjacency
     +---w (hello-adjacency-type)?
      |     +---: (targeted)
      |        +---w targeted!
      |        +---w target-address?  inet:ip-address
      +---: (link)
          +---w link!
          +---w next-hop-interface?  mpls-interface-ref
          +---w next-hop-address?  inet:ip-address
  +---x mpls-ldp-clear-peer-statistics
    +---w input
     +---w lsr-id?  union

Figure 12

4. mLDP YANG Model

4.1. Overview

Due to tight dependency of mLDP on LDP, mLDP model builds on top of LDP model defined earlier in the document. Following are the main mLDP areas and documents that are within the scope of this model:

- mLDP Base Specification [RFC6388]
- mLDP Recursive FEC [RFC6512]
- Targeted mLDP [RFC7060]
- mLDP Fast-Reroute (FRR)
  * Node Protection [RFC7715]
  * Multicast-only
- Hub-and-Spoke Multipoint LSPs [RFC7140]
- mLDP In-band Signaling [RFC6826] (future revision)
- mLDP In-band signaling in a VRF [RFC7246]
o mLDP In-band Signaling with Wildcards [RFC7438] (future revision)
o Configured Leaf LSPs (manually provisioned)

[Ed Note: Some of the topics in the above list are to be addressed/add in later revision of this document].

4.2. Configuration

4.2.1. Configuration Hierarchy

In terms of overall configuration layout, following figure highlights extensions to LDP configuration model to incorporate mLDP:
Figure 13

From above hierarchy, we can categorize mLDP configuration parameters into two types:
Parameters that leverage/extend LDP containers and parameters

Parameters that are mLDP specific

Following subsections first describe mLDP specific configuration parameters, followed by those leveraging LDP.

4.2.2. mldp container

mldp container resides directly under "mpls-ldp" and holds the configuration related to items that are mLDP specific. The main items under this container are:

- mLDP enabling: To enable mLDP under a (VRF) routing instance, mldp container is enabled under LDP. Given that mLDP requires LDP signalling, it is not sensible to allow disabling LDP control plane under a (VRF) network-instance while requiring mLDP to be enabled for the same. However, if a user wishes only to allow signalling for multipoint FECs on an LDP/mLDP enabled VRF instance, he/she can use LDP label-policies to disable unicast FECs under the VRF.

- mLDP per-AF features: mLDP manages its own list of IP address-families and the features enabled underneath. The per-AF mLDP configuration items include:
  
  - Multicast-only FRR: This enables Multicast-only FRR functionality for a given AF under mLDP. The feature allows route-policy to be configured for finer control/applicability of the feature.
  
  - Recursive FEC: The recursive-fec feature [RFC6512] can be enabled per AF with a route-policy.
  
  - Configured Leaf LSPs: To provision multipoint leaf LSP manually, a container is provided per-AF under LDP. The configuration is flexible and allows a user to specify MP LSPs of type p2mp or mp2mp with IPv4 or IPv6 root address(es) by using either LSP-Id or (S,G).

Targeted mLDP feature specification [RFC7060] do not require any mLDP specific configuration. It, however, requires LDP upstream-label-assignment capability [RFC6389] to be enabled.
4.2.3. Leveraging LDP containers

mLDP configuration model leverages following configuration areas and containers that are already defined for LDP:

- Capabilities: A new container "mldp" is defined under Capabilities container. This new container specifies any mLDP specific capabilities and their parameters. Moreover, a new "mldp" container is also added under per-peer capability container to override/control mLDP specific capabilities on a peer level. In the scope of this document, the most important capabilities related to mLDP are p2mp, mp2mp, make-before-break, hub-and-spoke, and node-protection.

- Discovery and Peer: mLDP requires LDP discovery and peer procedures to form mLDP peering. A peer is treated as mLDP peer only when either P2MP or MP2MP capabilities have been successfully exchanged with the peer. If a user wish to selectively enable or disable mLDP with a LDP-enabled peer, he/she may use per-peer mLDP capabilities configuration. [Ed Note: The option to control mLDP enabling/disabling on a peer-list is being explored for future]. In most common deployments, it is desirable to disable mLDP (capabilities announcements) on a targeted-only LDP peering, where targeted-only peer is the one whose discovery sources are targeted only. In future revision, a configuration option for this support will also be provided.

- Forwarding: By default, mLDP is allowed to select any of the LDP enabled interface as a downstream interface towards a nexthop (LDP/mLDP peer) for MP LSP programming. However, a configuration option is provided to allow mLDP to exclude a given interface from such a selection. Note that such a configuration option will be useful only when there are more than one interfaces available for the downstream selection.

This goes without saying that mLDP configuration tree follows the same approach as LDP, where the tree comprise leafs for intended configuration.

4.2.4. YANG tree

The following figure captures the YANG tree for mLDP configuration. To keep the focus, the figure has been simplified to display only mLDP items without any LDP items.
<pre>++--rw global
++--rw config
  ++--rw capability
    ++--rw mldp (mldp)?
      ++--rw p2mp
        | ++--rw enable? boolean
      ++--rw mp2mp
        | ++--rw enable? boolean
      ++--rw make-before-break
        | ++--rw enable? boolean
        | ++--rw switchover-delay? uint16
        | ++--rw timeout? uint16
      ++--rw hub-and-spoke (capability-mldp-hsmp)?
        | ++--rw enable? boolean
      ++--rw node-protection (capability-mldp-node-protection)?
        | ++--rw plr? boolean
        | ++--rw merge-point
        | ++--rw enable? boolean
        | ++--rw targeted-session-teardown-delay? uint16
    ++--rw mldp (mldp)?
      ++--rw config
        | ++--rw enable? boolean
      ++--rw address-family* ([afi])
        ++--rw afi ldp-address-family
          ++--rw config
            | ++--rw multicast-only-frr (mldp-mofrr)?
            | | ++--rw prefix-list? prefix-list-ref
            | ++--rw recursive-fec
            | ++--rw prefix-list? prefix-list-ref
          ++--rw configured-leaf-lsp
      ++--rw p2mp
        | ++--rw roots-ipv4
        | | ++--rw root* [root-address]
        | | | ++--rw root-address inet:ipv4-address
        | | | ++--rw lsp* [lsp-id source-address group-address]
        | | | | ++--rw lsp-id uint16
        | | | | ++--rw source-address inet:ipv4-address
        | | | | ++--rw group-address inet:ipv4-address-no-zone
        | ++--rw roots-ipv6
        | | ++--rw root* [root-address]
        | | | ++--rw root-address inet:ipv6-address
        | | | ++--rw lsp* [lsp-id source-address group-address]
        | | | | ++--rw lsp-id uint16
        | | | | ++--rw source-address inet:ipv6-address
        | | | | ++--rw group-address inet:ipv6-address-no-zone
      ++--rw mp2mp
        | ++--rw roots-ipv4
          | | ++--rw root* [root-address]
</pre>
4.3. Operational State

Operational state of mLDP can be queried and obtained from this read-only container "mldp" which resides under mpls-ldp container.
Please note this state tree refers both the configuration "applied" state as well as the "derived" state related to the mLDP protocol.

Following is a simplified graphical representation of the data model for mLDP operational state:

```yaml
module: ietf-mpls-ldp
augment /rt:routing/rt:control-plane-protocols:
  +++rw mpls-ldp!
  +++rw global
  +++ro state
    +++ro capability
    +++ro mldp (mldp)?
    |  +++ro p2mp
    |    |  +++ro enable? boolean
    |  +++ro mp2mp
    |    |  +++ro enable? boolean
    |  +++ro make-before-break
    |    |  +++ro enable? boolean
    |    |  +++ro switchover-delay? uint16
    |    |  +++ro timeout? uint16
    |  +++ro hub-and-spoke {capability-mldp-hsmp}?  
    |    |  +++ro enable? boolean
    |  +++ro node-protection {capability-mldp-node-protection}?
    |    |  +++ro plr? boolean
    |    |  +++ro merge-point
    |    |    |  +++ro enable? boolean
    |    |    |  +++ro targeted-session-teardown-delay? uint16
    |  +++ro mldp (mldp)?
    |  +++ro enable? boolean
    |  +++rw address-family* [afi]
    |    |  +++rw afi ldp-address-family
    |    |  +++ro state
    |    |    |  +++ro multicast-only-frr (mldp-mofrr)?
    |    |    |    |  +++ro prefix-list? prefix-list-ref
    |    |    |  +++ro recursive-fec
    |    |    |  +++ro prefix-list? prefix-list-ref
    |    |    |  +++ro ipv4
    |    |    |    |  +++ro roots
    |    |    |    |    |  +++ro root* [root-address]
    |    |    |    |    |    |  +++ro root-address inet:ipv4-address
    |    |    |    |    |    |  +++ro is-self? boolean
    |    |    |    |    |  +++ro reachability* [address interface]
    |    |    |    |    |    |  +++ro address inet:ipv4-address
    |    |    |    |    |    |  +++ro interface mpls-interface-ref
```
++-ro ipv6
   +++-ro roots
      +++-ro root* [root-address]
         +++-ro root-address inet:ipv6-address
         +++-ro is-self? boolean
         +++-ro reachability* [address interface]
            +++-ro address inet:ipv6-address
            +++-ro interface mpls-interface-ref
            +++-ro peer? leafref
   +++-ro bindings
      +++-ro opaque-type-lspid
         +++-ro fec-label* [root-address lsp-id]
            +++-ro opaque-type-lspid
               +++-ro root-address inet:ipv6-address
               +++-ro lsp-id uint32
               +++-ro recur-root-address inet:ipv6-address
               +++-ro recur-rd route-distinguisher
               +++-ro multipoint-type? multipoint-type
               +++-ro peer* [direction peer advertisement-type]
                  +++-ro direction downstream-upstream
                  +++-ro peer leafref
                  +++-ro advertisement-type advertised-received
                  +++-ro label? mpls:mpls-label
                  +++-ro mbb-role? enumeration
                  +++-ro mofrr-role? enumeration
            +++-ro opaque-type-src
               +++-ro fec-label* [root-address source-address]
                  +++-ro opaque-type-src
                     +++-ro root-address inet:ipv6-address
                     +++-ro source-address inet:ipv6-address
                     +++-ro group-address inet:ipv6-address
                     +++-ro rd route-distinguisher
                  +++-ro opaque-type-bidir
                     +++-ro fec-label* [root-address rp group-address]
                        +++-ro opaque-type-bidir
                           +++-ro root-address inet:ipv6-address
                           +++-ro rp inet:ipv6-address
                           +++-ro group-address inet:ipv6-address
                           +++-ro rd route-distinguisher
                           +++-ro recur-root-address inet:ipv6-address
                           +++-ro recur-rd route-distinguisher

---ro multipoint-type?       multipoint-type
---ro peer* [direction peer advertisement-type]
  ---ro direction                downstream-upstream
  ---ro peer                    leafref
  ---ro advertisement-type      advertised-received
  ---ro label?                  mpls:mpls-label
  ---ro mbb-role?               enumeration
  ---ro mofrr-role?             enumeration
---rw forwarding-nexthop {forwarding-nexthop-config}?
  ---rw interfaces
    ---rw interface* [interface]
      ---rw address-family* [afi]
      ---ro state
        ---ro mldp-disable? boolean {mldp}?
---rw peers
  ---rw peer* [lsr-id]
  ---ro state
    ---ro capability
      ---ro mldp {mldp}?
        ---ro p2mp
          ---ro enable? boolean
        ---ro mp2mp
          ---ro enable? boolean
          ---ro make-before-break
            ---ro enable? boolean
            ---ro switchover-delay? uint16
            ---ro timeout? uint16
            ---ro hub-and-spoke {capability-mldp-hsmp}?
              ---ro enable? boolean
        ---ro node-protection {capability-mldp-node-protection}?
          ---ro plr? boolean
          ---ro merge-point
            ---ro enable? boolean
            ---ro targeted-session-teardown-delay? uint16
---ro received-peer-state
  ---ro capability
    ---ro mldp {mldp}?
      ---ro p2mp
        ---ro enable? boolean
      ---ro mp2mp
        ---ro enable? boolean
        ---ro make-before-break
          ---ro enable? boolean
        ---ro hub-and-spoke
          ---ro enable? boolean
        ---ro node-protection
          ---ro plr? boolean
          ---ro merge-point? boolean
4.3.1. Derived states

Following are main areas for which mLDP operational derived state is defined:

- Root
- Bindings (FEC-label)
- Capabilities

4.3.1.1. Root state

Root address is a fundamental construct for MP FEC bindings and LSPs. The root state provides information on all the known roots in a given address-family, and their information on the root reachability (as learnt from RIB). In case of multi-path reachability to a root, the selection of upstream path is done on per-LSP basis at the time of LSP setup. Similarly, when protection mechanisms like MBB or MoFRR are in place, the path designation as active/standby or primary/backup is also done on per LSP basis. It is to be noted that a given root can be shared amongst multiple P2MP and/or MP2MP LSPs. Moreover, an LSP can be signaled to more than one root for RNR purposes.

The following diagram illustrates a root database on a branch/transit LSR:
root 1.1.1.1:
  path1:
    RIB: GigEthernet 1/0, 12.1.0.2;
    LDP: peer 192.168.0.1:0
  path2:
    RIB: GigEthernet 2/0, 12.2.0.2;
    LDP: peer 192.168.0.3:0

root 2.2.2.2:
  path1:
    RIB: 3.3.3.3;       (NOTE: This is a recursive path)
    LDP: peer 192.168.0.3:0  (NOTE: T-mLDP peer)

root 9.9.9.9:
  . . .

Figure 16
A root entry on a root LSR itself will be presented as follows:

root 9.9.9.9:
  is-self

Figure 17

4.3.1.2. Bindings state

Binding state provides information on mLDP FEC-label bindings for both P2MP and MP2MP FEC types. Like LDP, the FEC-label binding derived state is presented in a FEC-centric view per address-family, and provides information on both inbound (received) and outbound (advertised) bindings. The FEC is presented as (root-address, opaque-type-data) and the direction (upstream or downstream) is picked with respect to root reachability. In case of MBB or/and MoFRR, the role of a given peer binding is also provided with respect to MBB (active or standby) or/and MoFRR (primary or backup).

This document covers following type of opaque values with their keys in the operational model of mLDP bindings:
<table>
<thead>
<tr>
<th>Opaque Type</th>
<th>Key</th>
<th>RFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic LSP Identifier</td>
<td>LSP Id</td>
<td>[RFC6388]</td>
</tr>
<tr>
<td>Transit IPv4 Source</td>
<td>Source, Group</td>
<td>[RFC6826]</td>
</tr>
<tr>
<td>Transit IPv6 Source</td>
<td>Source, Group</td>
<td>[RFC6826]</td>
</tr>
<tr>
<td>Transit IPv4 Bidir</td>
<td>RP, Group</td>
<td>[RFC6826]</td>
</tr>
<tr>
<td>Transit IPv6 Bidir</td>
<td>RP, Group</td>
<td>[RFC6826]</td>
</tr>
<tr>
<td>Transit VPNv4 Source</td>
<td>Source, Group, RD</td>
<td>[RFC7246]</td>
</tr>
<tr>
<td>Transit VPNv6 Source</td>
<td>Source, Group, RD</td>
<td>[RFC7246]</td>
</tr>
<tr>
<td>Transit VPNv4 Bidir</td>
<td>RP, Group, RD</td>
<td>[RFC7246]</td>
</tr>
<tr>
<td>Transit VPNv6 Bidir</td>
<td>RP, Group, RD</td>
<td>[RFC7246]</td>
</tr>
<tr>
<td>Recursive Opaque</td>
<td>Root</td>
<td>[RFC6512]</td>
</tr>
<tr>
<td>VPN-Recursive Opaque</td>
<td>Root, RD</td>
<td>[RFC6512]</td>
</tr>
</tbody>
</table>

Table 1: MP Opaque Types and keys

It is to be noted that there are three basic types (LSP Id, Source, and Bidir) and then there are variants (VPN, recursive, VPN-recursive) on top of these basic types.

Following captures high level tree hierarchy for mLDP bindings state:

```yang
++--rw mpls-ldp!
  ++--rw mldp
    ++--rw address-family* [afi]
      ++--rw afi address-family
        ++--ro state
          ++--ro bindings
            ++--ro opaque-type-xxx [root-address, type-specific-key]
              ++--ro root-address
              ++--ro ...
              ++--ro recur-root-address inet:ipv4-address
              ++--ro recur-rd route-distinguisher
              ++--ro multipoint-type? multipoint-type
              ++--ro peer* [direction peer advertisement-type]
                ++--ro direction downstream-upstream
                ++--ro peer leafref
                ++--ro advertisement-type advertised-received
                ++--ro label? mpls:mpls-label
                ++--ro mbbr-role? enumeration
                ++--ro mofrr-role? enumeration

Figure 18
```

In the above tree, the type-specific-key varies with the base type as listed in earlier Table 1. For example, if the opaque type is Generic LSP Identifier, then the type-specific-key will be a uint32 value corresponding to the LSP. Please see the complete model for all other types.

Moreover, the binding tree defines only three types of sub-trees (i.e. lspid, src, and bidir) which is able to map the respective variants (vpn, recursive, and vpn-recursive) accordingly. For example, the key for opaque-type-src is \([R, S, G, rd, recur-R, recur-RD]\), where basic type will specify \([R, S,G,-, -, -]\), VPN type will specify \([R, S,G, rd, -, -]\), recursive type will specify \([R, S,G, -, recur-R, recur-R, recur-rd]\) and VPN-recursive type will specify \([R, S,G, -, recur-R, recur-rd]\).

It is important to take note of the following:

- The address-family ipv4/ipv4 applies to "root" address in the mLDP binding tree. The other addresses (source, group, RP etc) do not have to be of the same address family type as the root.

- The "recur-root-address" field applies to Recursive opaque type, and (recur-root-address, recur-rd) fields applies to VPN-Recursive opaque types as defined in [RFC6512]

- In case of a recursive FEC, the address-family of the recur-root-address could be different than the address-family of the root address of original encapsulated MP FEC

The following diagram illustrates the FEC-label binding information structure for a P2MP (Transit IPv4 Source type) LSP on a branch/transit LSR:

```
FEC (root 2.2.2.2, S=192.168.1.1, G=224.1.1.1):
  type: p2mp
  upstream:
    advertised:
      peer 192.168.0.1:0, label 16000 (local)
  downstream:
    received:
      peer 192.168.0.2:0, label 17000 (remote)
      peer 192.168.0.3:0, label 18000 (remote)
```

Figure 19
The following diagram illustrates the FEC-label binding information structure for a similar MP2MP LSP on a branch/transit LSR:

FEC (root 2.2.2.2, RP=192.168.9.9, G=224.1.1.1):
  type: mp2mp
  upstream:
    advertised:
    peer 192.168.0.1:0, label 16000 (local)
    received:
    peer 192.168.0.1:0, label 17000 (remote)
  downstream:
    advertised:
    peer 192.168.0.2:0, label 16001 (local), MBB role=active
    peer 192.168.0.3:0, label 16002 (local), MBB role=standby
    received:
    peer 192.168.0.2:0, label 17001 (remote)
    peer 192.168.0.3:0, label 18001 (remote)

Figure 20

4.3.1.3. Capabilities state

Like LDP, mLDP capabilities state comprise two types of information - global information and per-peer information.

4.4. Notifications

mLDP notification module consists of notification related to changes in the operational state of an mLDP FEC. Following is a simplified graphical representation of the data model for mLDP notifications:

```
  notifications:
    +----n mpls-mldp-fec-event
      +----ro event-type?     oper-status-event-type
      +----ro tree-type?      multipoint-type
      +----ro root?           inet:ip-address
      +----ro (lsp-key-type)?
        +----(lsp-id-based)
          |   +----ro lsp-id?     uint16
        +----(source-group-based)
          +----ro source-address? inet:ip-address
          +----ro group-address? inet:ip-address
```

Figure 21
4.5. Actions

Currently, no RPCs/actions are defined for mLDP.

5. Open Items

Following is a list of open items that are to be discussed and addressed in future revisions of this document:

- Close on augmentation off "mpls" list in "ietf-mpls" defined in [I-D.saad-mpls-base-yang]
- Align operational state modeling with other routing protocols and [I-D.openconfig-netmod-opstate]
- Complete the section on Protocol-centric implementations and all-vrfs
- Specify default values for configuration parameters
- Revisit and cut down on the scope of the document and number of features it is trying to cover
- Split the model into a base and extended items
- Add statistics for mLDP root LSPs and bindings
- Extend the "Configured Leaf LSPs" for various types of opaque-types
- Extend mLDP notifications for other types of opaque values as well
- Close on single vs separate document for mLDP Yang

6. YANG Specification

Following are actual YANG definition for LDP and mLDP constructs defined earlier in the document.

<CODE BEGINS> file "ietf-mpls-ldp@2016-07-08.yang" -->

module ietf-mpls-ldp {
    namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-ldp";
    // replace with IANA namespace when assigned
    prefix ldp;
    import ietf-inet-types {

prefix "inet";
}

import ietf-yang-types {
    prefix "yang";
}

import ietf-interfaces {
    prefix "if";
}

import ietf-ip {
    prefix "ip";
}

import ietf-routing {
    prefix "rt";
}

import ietf-mpls {
    prefix "mpls";
}

organization
    "IETF MPLS Working Group";

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

    WG Chair: Loa Andersson
            <mailto:loa@pi.nu>

    WG Chair: Ross Callon
            <mailto:rcallon@juniper.net>

    WG Chair: George Swallow
            <mailto:swallow.ietf@gmail.com>

    Editor: Kamran Raza
            <mailto:skraza@cisco.com>

    Editor: Rajiv Asati
            <mailto:rajiva@cisco.com>

    Editor: Xufeng Liu
            <mailto:xliu@kuatrotech.com>

    Editor: Santosh Esale
This YANG module defines the essential components for the management of Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP) and Multipoint LDP (mLDP).
feature capability-mldp-node-protection {
  description
    "This feature indicates that the system allows to configure
    mLDP node-protection capability.";
}

feature capability-typed-wildcard-fec {
  description
    "This feature indicates that the system allows to configure
    LDP typed-wildcard-fec capability.";
}

feature capability-upstream-label-assignment {
  description
    "This feature indicates that the system allows to configure
    LDP upstream label assignment capability.";
}

feature forwarding-nexthop-config {
  description
    "This feature indicates that the system allows to configure
    forwarding nexthop on interfaces.";
}

feature global-session-authentication {
  description
    "This feature indicates that the system allows to configure
    authentication at global level.";
}

feature graceful-restart-helper-mode {
  description
    "This feature indicates that the system supports graceful
    restart helper mode.";
}

feature mldp {
  description
    "This feature indicates that the system supports Multicast
    LDP (mLDP).";
}

feature mldp-mofrr {
  description
    "This feature indicates that the system supports mLDP
    Multicast only FRR (MoFRR).";
}
feature per-interface-timer-config {
  description
    "This feature indicates that the system allows to configure
    interface hello timers at the per-interface level.";
}

feature per-peer-graceful-restart-config {
  description
    "This feature indicates that the system allows to configure
    graceful restart at the per-peer level.";
}

feature per-peer-session-attributes-config {
  description
    "This feature indicates that the system allows to configure
    session attributes at the per-peer level.";
}

feature policy-extended-discovery-config {
  description
    "This feature indicates that the system allows to configure
    policies to control the acceptance of extended neighbor
discovery hello messages.";
}

feature policy-label-assignment-config {
  description
    "This feature indicates that the system allows to configure
    policies to assign labels according to certain prefixes.";
}

feature policy-ordered-label-config {
  description
    "This feature indicates that the system allows to configure
    ordered label policies.";
}

feature session-downstream-on-demand-config {
  description
    "This feature indicates that the system allows to configure
    session downstream-on-demand";
}

/*
 * Typedefs
 */
typedef ldp-address-family {
  type identityref {

base rt:address-family;
} description
"LDP address family type."

typedef duration32-inf {
type union {
type uint32;
type enumeration {
enum "infinite" {
description "The duration is infinite.";
}
}
} units seconds;
description
"Duration represented as 32 bit seconds with infinite.";

typedef advertised-received {
type enumeration {
enum advertised {
description "Advertised information.";
}
enum received {
description "Received information.";
}
} description
"Received or advertised."

typedef downstream-upstream {
type enumeration {
enum downstream {
description "Downstream information.";
}
enum upstream {
description "Upstream information.";
}
} description
"Received or advertised."

typedef label-adv-mode {
type enumeration {

enum downstream-unsolicited {
    description "Downstream Unsolicited.";
}
enum downstream-on-demand {
    description "Downstream on Demand.";
}

description
"Label Advertisement Mode.";

typedef mpls-interface-ref {
    type leafref {
        path "/rt:routing/mpls:mpls/mpls:interface/mpls:name";
    }
    description
    "This type is used by data models that need to reference mpls interfaces.";
}

typedef multipoint-type {
    type enumeration {
        enum p2mp {
            description "Point to multipoint.";
        }
        enum mp2mp {
            description "Multipoint to multipoint.";
        }
    }
    description
    "p2mp or mp2mp.";
}

typedef neighbor-list-ref {
    type string;
    description
    "A type for a reference to a neighbor list.";
}

typedef peer-list-ref {
    type string;
    description
    "A type for a reference to a peer list.";
}

typedef prefix-list-ref {
    type string;
    description

"A type for a reference to a prefix list."
}

typedef oper-status-event-type {
  type enumeration {
    enum up {
      value 1;
      description
      "Operational status changed to up.";
    }
    enum down {
      value 2;
      description
      "Operational status changed to down.";
    }
  }
  description "Operational status event type for notifications.";
}

typedef route-distinguisher {
  type string {
  }
  description
  "Type definition for route distinguisher.";
  reference
  "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
}

/*
 * Identities
 */
identity adjacency-flag-base {
  description "Base type for adjacency flags.";
}

identity adjacency-flag-active {
  base "adjacency-flag-base";
  description
  "This adjacency is configured and actively created.";
}

identity adjacency-flag-passive {
  base "adjacency-flag-base";
  description
  "This adjacency is not configured and passively accepted.";
}

/*
/* Groupings */

grouping adjacency-state-attributes {
  description
    "Adjacency state attributes.";

  leaf-list flag {
    type identityref {
      base "adjacency-flag-base";
    }
    description "Adjacency flags.";
  }

  container hello-holdtime {
    description "Hello holdtime state.";
    leaf adjacent {
      type uint16;
      units seconds;
      description "Peer holdtime.";
    }
    leaf negotiated {
      type uint16;
      units seconds;
      description "Negotiated holdtime.";
    }
    leaf remaining {
      type uint16;
      units seconds;
      description "Remaining holdtime.";
    }
  }

  leaf next-hello {
    type uint16;
    units seconds;
    description "Time to send the next hello message.";
  }

  container statistics {
    description
      "Statistics objects.";

    leaf discontinuity-time {
      type yang:date-and-time;
      mandatory true;
      description
        "The time on the most recent occasion at which any one or
         more of this interface’s counters suffered a
discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem re-initialized itself.

leaf hello-received {
    type yang:counter64;
    description
    "The number of hello messages received.";
}
leaf hello-dropped {
    type yang:counter64;
    description
    "The number of hello messages received.";
}

} // statistics
} // adjacency-state-attributes

grouping basic-discovery-timers {
    description
    "Basic discovery timer attributes.";
    leaf hello-holdtime {
        type uint16 {
            range 15..3600;
        }
        units seconds;
        description
        "The time interval for which a LDP link Hello adjacency is maintained in the absence of link Hello messages from the LDP neighbor";
    }
    leaf hello-interval {
        type uint16 {
            range 5..1200;
        }
        units seconds;
        description
        "The interval between consecutive LDP link Hello messages used in basic LDP discovery";
    }
} // basic-discovery-timers

grouping binding-address-state-attributes {
    description
    "Address binding attributes";
    leaf advertisement-type {
        type advertised-received;
leaf peer {
  type leafref {
    path "../../../../../peers/peer/lsr-id";
  }
  must ".../advertisement-type = 'received'" {
    description
    "Applicable for received address."
  }
  description
  "LDP peer from which this address is received.";
} // peer
} // binding-address-state-attributes

grouping binding-label-state-attributes {
  description
  "Label binding attributes";
  list peer {
    key "peer advertisement-type";
    description
    "List of advertised and received peers.";
    leaf peer {
      type leafref {
        path "../../../../../peers/peer/lsr-id";
      }
      description
      "LDP peer from which this binding is received,
      or to which this binding is advertised.";
    }
    leaf advertisement-type {
      type advertised-received;
      description
      "Received or advertised.";
    }
    leaf label {
      type mpls:mpls-label;
      description
      "Advertised (outbound) or received (inbound)
      label.";
    }
    leaf used-in-forwarding {
      type boolean;
      description
      "true' if the label is used in forwarding.";
    }
  } // peer
grouping extended-discovery-policy-attributes {
  description
    "LDP policy to control the acceptance of extended neighbor
discovery hello messages."
  container hello-accept {
    if-feature policy-extended-discovery-config;
    description
      "Extended discovery acceptance policies."
    leaf enable {
      type boolean;
      description
        "'true' to accept; 'false' to deny.";
    }
    leaf neighbor-list {
      type neighbor-list-ref;
      description
        "The name of a peer ACL.";
    }
  }
  // hello-accept
// extended-discovery-policy-attributes

  grouping extended-discovery-timers {
    description
      "Extended discovery timer attributes.";
    leaf hello-holdtime {
      type uint16 {
        range 15..3600;
      }
      units seconds;
      description
        "The time interval for which LDP targeted Hello adjacency
is maintained in the absence of targeted Hello messages
from an LDP neighbor.";
    }
    leaf hello-interval {
      type uint16 {
        range 5..3600;
      }
      units seconds;
      description
        "The interval between consecutive LDP targeted Hello
messages used in extended LDP discovery.";
    }
  }
} // binding-label-state-attributes
grouping global-attributes {
  description "Configuration attributes at global level.";
  uses instance-attributes;
} // global-attributes

grouping graceful-restart-attributes {
  description "Graceful restart configuration attributes.";
  container graceful-restart {
    description "Attributes for graceful restart.";
    leaf enable {
      type boolean;
      description "Enable or disable graceful restart.";
    }
    leaf helper-enable {
      if-feature graceful-restart-helper-mode;
      type boolean;
      description "Enable or disable graceful restart helper mode.";
    }
    leaf reconnect-time {
      type uint16 {
        range 10..1800;
      }
      units seconds;
      description "Specifies the time interval that the remote LDP peer must wait for the local LDP peer to reconnect after the remote peer detects the LDP communication failure.";
    }
    leaf recovery-time {
      type uint16 {
        range 30..3600;
      }
      units seconds;
      description "Specifies the time interval, in seconds, that the remote LDP peer preserves its MPLS forwarding state after receiving the Initialization message from the restarted local LDP peer.";
    }
    leaf forwarding-holdtime {
      type uint16 {

range 30..3600;
} // graceful-restart-attributes

grouping graceful-restart-attributes-per-peer {
  description "Per peer graceful restart configuration attributes.";
  container graceful-restart {
    description "Attributes for graceful restart.";
    leaf enable {
      type boolean;
      description "Enable or disable graceful restart.";
    }
    leaf reconnect-time {
      type uint16 {
        range 10..1800;
      }
      units seconds;
      description "Specifies the time interval that the remote LDP peer must wait for the local LDP peer to reconnect after the remote peer detects the LDP communication failure.";
    }
    leaf recovery-time {
      type uint16 {
        range 30..3600;
      }
      units seconds;
      description "Specifies the time interval, in seconds, that the remote LDP peer preserves its MPLS forwarding state after receiving the Initialization message from the restarted local LDP peer.";
    }
  }
} // graceful-restart

grouping instance-attributes {
  description "Configuration attributes at instance level.";
}
container capability {
  description "Configure capability.";
  container end-of-lib {
    if-feature capability-end-of-lib;
    description
      "Configure end-of-lib capability.";
    leaf enable {
      type boolean;
      description
        "Enable end-of-lib capability.";
    }
  }
  container typed-wildcard-fec {
    if-feature capability-typed-wildcard-fec;
    description
      "Configure typed-wildcard-fec capability.";
    leaf enable {
      type boolean;
      description
        "Enable typed-wildcard-fec capability.";
    }
  }
  container upstream-label-assignment {
    if-feature capability-upstream-label-assignment;
    description
      "Configure upstream label assignment capability.";
    leaf enable {
      type boolean;
      description
        "Enable upstream label assignment.";
    }
  }
  container mldp {
    if-feature mldp;
    description
      "Multipoint capabilities.";
    uses mldp-capabilities;
  }
}

// capability

uses graceful-restart-attributes;

leaf igp-synchronization-delay {
  type uint16 {
    range 3..60;
  }
  units seconds;
description
"Sets the interval that the LDP waits before notifying the
Interior Gateway Protocol (IGP) that label exchange is
completed so that IGP can start advertising the normal
metric for the link.";

leaf lsr-id {
  type yang:dotted-quad;
  description "Router ID.";
}

} // instance-attributes

grouping ldp-adjacency-ref {
  description
  "An absolute reference to an LDP adjacency.";
  choice hello-adjacency-type {
    description
    "Interface or targeted adjacency.";
    case targeted {
      container targeted {
        description "Targeted adjacency.";
        leaf target-address {
          type inet:ip-address;
          description
          "The target address.";
        }
      } // targeted
    }
    case link {
      container link {
        description "Link adjacency.";
        leaf next-hop-interface {
          type mpls-interface-ref;
          description
          "Interface connecting to next-hop.";
        }
        leaf next-hop-address {
          type inet:ip-address;
          must "./next-hop-interface" {
            description
            "Applicable when interface is specified.";
          }
          description
          "IP address of next-hop.";
        }
      } // link
    }
  }
} // ldp-adjacency-ref
grouping ldp-fec-event {
  description
    "A LDP FEC event.";
  leaf prefix {
    type inet:ip-prefix;
    description
      "FEC.";
  }
}

grouping ldp-peer-ref {
  description
    "An absolute reference to an LDP peer.";
  leaf peer-ref {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/mpls-ldp/
           + "peers/peer/lsr-id";
    }
    description
      "Reference to an LDP peer."
  }
}

grouping mldp-capabilities {
  description
    "mLDP capabilities.";
  container p2mp {
    description
      "Configure point-to-multipoint capability.";
    leaf enable {
      type boolean;
      description
        "Enable point-to-multipoint."
    }
  }
  container mp2mp {
    description
      "Configure multipoint-to-multipoint capability.";
    leaf enable {
      type boolean;
      description
        "Enable multipoint-to-multipoint."
    }
  }
  container make-before-break {
  }
}
description
  "Configure make-before-break capability.";
leaf enable {
  type boolean;
  description
    "Enable make-before-break.";
}
leaf switchover-delay {
  type uint16;
  units seconds;
  description
    "Switchover delay in seconds.";
}
leaf timeout {
  type uint16;
  units seconds;
  description
    "Timeout in seconds.";
}
}
container hub-and-spoke {
  if-feature capability-mldp-hsmp;
  description
    "Configure hub-and-spoke-multipoint capability.";
  reference
    "RFC7140: LDP Extensions for Hub and Spoke Multipoint
     Label Switched Path";
  leaf enable {
    type boolean;
    description
      "Enable hub-and-spoke-multipoint.";
  }
}
container node-protection {
  if-feature capability-mldp-node-protection;
  description
    "Configure node-protection capability.";
  reference
    "RFC7715: mLDP Node Protection.";
  leaf plr {
    type boolean;
    description
      "Point of Local Repair capable for MP LSP node
       protection.";
  }
  container merge-point {
    description
      "Merge Point capable for MP LSP node protection.";
  }
}
leaf enable {
    type boolean;
    description
        "Enable merge point capability.";
}
leaf targeted-session-teardown-delay {
    type uint16;
    units seconds;
    description
        "Targeted session teardown delay.";
}
// merge-point
}
// mldp-capabilities

grouping mldp-configured-lsp-roots {
    description
        "mLDP roots containers.";
    container roots-ipv4 {
        when ".//.../af = ‘ipv4’" {
            description
                "Only for IPv4.";
        }
        description
            "Configured IPv4 multicast LSPs."
        list root {
            key "root-address";
            description
                "List of roots for configured multicast LSPs.";
            leaf root-address {
                type inet:ipv4-address;
                description
                    "Root address.";
            }
        }
        list lsp {
            must "((lsp-id = 0 and source-address != ‘0.0.0.0’ and "
                + "group-address != ‘0.0.0.0’) or "
                + "(lsp-id != 0 and source-address = ‘0.0.0.0’ and "
                + "group-address = ‘0.0.0.0’)" {
                description
                    "A LSP can be identified by either <lsp-id> or "
                    + "<source-address, group-address>.";
            }
            key "lsp-id source-address group-address";
        }
    }
}

description "List of LSPs.";
leaf lsp-id {
  type uint16;
  description "ID to identify the LSP.";
}
leaf source-address {
  type inet:ipv4-address;
  description "Source address.";
}
leaf group-address {
  type inet:ipv4-address-no-zone;
  description "Group address.";
}
} // list lsp
} // list root
} // roots-ipv4

container roots-ipv6 {
  when "../../../af = 'ipv6'" {
    description "Only for IPv6.";
  }
  description "Configured IPv6 multicast LSPs."
}

list root {
  key "root-address";
  description "List of roots for configured multicast LSPs.";
  leaf root-address {
    type inet:ipv6-address;
    description "Root address.";
  }
}

list lsp {
  must "(lsp-id = 0 and source-address != '::' and " + "group-address != '::') or " + "(lsp-id != 0 and source-address = '::' and " + "group-address = '::')" {
    description "A LSP can be identified by either <lsp-id> or <source-address, group-address>.";
  }
key "lsp-id source-address group-address";
description
"List of LSPs.";
leaf lsp-id {
  type uint16;
  description "ID to identify the LSP.";
}
leaf source-address {
  type inet:ipv6-address;
  description
  "Source address.";
}
leaf group-address {
  type inet:ipv6-address-no-zone;
  description
  "Group address.";
}
} // list lsp
} // list root
} // roots-ipv6
} // mldp-configured-lsp-roots

grouping mldp-fec-event {
  description
  "A mLDP FEC event.";
  leaf tree-type {
    type multipoint-type;
    description
    "p2mp or mp2mp.";
  }
  leaf root {
    type inet:ip-address;
    description
    "Root address.";
  }
  choice lsp-key-type {
    description
    "LSP ID based or source-group based .";
    case lsp-id-based {
      leaf lsp-id {
        type uint16;
        description
        "ID to identify the LSP.";
      }
    }
    case source-group-based {
      leaf source-address {
type inet:ip-address;
description
  "LSP source address."
}
leaf group-address {
  type inet:ip-address;
  description
  "Multicast group address."
}
} // case source-group-based
} // mldp-fec-event

grouping mldp-binding-label-state-attributes {
  description
  "mLDP label binding attributes."
  leaf multipoint-type {
    type multipoint-type;
    description
    "The type of multipoint, p2mp or mp2mp."
  }
  list peer {
    key "direction peer advertisement-type";
    description
    "List of advertised and received peers."
    leaf direction {
      type downstream-upstream;
      description
      "Downstream or upstream."
    }
    leaf peer {
      type leafref {
        path
        "../.../../.../../.../../.../../.../peers/peer/lsr-id";
      }
      description
      "LDP peer from which this binding is received, or to which this binding is advertised."
    }
    leaf advertisement-type {
      type advertised-received;
      description
      "Advertised or received."
    }
    leaf label {
      type mpls:mpls-label;
    }
  }
} // mldp-binding
description
"Advertised (outbound) or received (inbound) label."
}
leaf mbb-role {
  when "../direction = 'upstream'" {
    description
    "For upstream."
  }
  type enumeration {
    enum none {
      description "MBB is not enabled.";
    }
    enum active {
      description "This LSP is active.";
    }
    enum inactive {
      description "This LSP is inactive.";
    }
  }
  description
  "The MBB status of this LSP.";
}
leaf mofrr-role {
  when "../direction = 'upstream'" {
    description
    "For upstream."
  }
  type enumeration {
    enum none {
      description "MOFRR is not enabled.";
    }
    enum primary {
      description "This LSP is primary.";
    }
    enum backup {
      description "This LSP is backup.";
    }
  }
  description
  "The MOFRR status of this LSP.";
}
} // peer
} // mldp-binding-label-state-attributes

grouping peer-af-policy-container {
  description
  "LDP policy attribute container under peer address-family.";
  container label-policy {
description "Label policy attributes.";
container advertise {
  description "Label advertising policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description "Applies the prefix list to outgoing label advertisements.";
  }
}
}
}

container accept {
  description "Label advertisement acceptance policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description "Applies the prefix list to incoming label advertisements.";
  }
}
}
}
}

grouping peer-attributes {
  description "Peer configuration attributes.";
  leaf session-ka-holdtime {
    type uint16 {
      range 45..3600;
    }
    units seconds;
    description "The time interval after which an inactive LDP session terminates and the corresponding TCP session closes. Inactivity is defined as not receiving LDP packets from the peer.";
  }
  leaf session-ka-interval {
    type uint16 {
      range 15..1200;
    }
    units seconds;
    description "The interval between successive transmissions of keepalive packets. Keepalive packets are only sent in the absence of other LDP packets transmitted over the LDP session.";
  }
}
grouping peer-authentication {
  description "Peer authentication attributes.";
  leaf session-authentication-md5-password {
    type string {
      length "1..80";
    }
    description "Assigns an encrypted MD5 password to an LDP peer";
  } // md5-password
} // peer-authentication

grouping peer-state-derived {
  description "Peer derived state attributes.";
  container label-advertisement-mode {
    description "Label advertisement mode state.";
    leaf local {
      type label-adv-mode;
      description "Local Label Advertisement Mode.";
    }
    leaf peer {
      type label-adv-mode;
      description "Peer Label Advertisement Mode.";
    }
    leaf negotiated {
      type label-adv-mode;
      description "Negotiated Label Advertisement Mode.";
    }
  }
  leaf next-keep-alive {
    type uint16;
    units seconds;
    description "Time to send the next KeepAlive message.";
  }
  leaf peer-ldp-id {
    type yang:dotted-quad;
    description "Peer LDP ID.";
  }
}
container received-peer-state {
  description "Peer features.";
  
  uses graceful-restart-attributes-per-peer;

  container capability {
    description "Configure capability.";
    container end-of-lib {
      description "Configure end-of-lib capability.";
      leaf enable {
        type boolean;
        description "Enable end-of-lib capability.";
      }
    }

    container typed-wildcard-fec {
      description "Configure typed-wildcard-fec capability.";
      leaf enable {
        type boolean;
        description "Enable typed-wildcard-fec capability.";
      }
    }

    container upstream-label-assignment {
      description "Configure upstream label assignment capability.";
      leaf enable {
        type boolean;
        description "Enable upstream label assignment.";
      }
    }

    container mldp {
      if-feature mldp;
      description "Multipoint capabilities.";

      container p2mp {
        description "Configure point-to-multipoint capability.";
        leaf enable {
          type boolean;
          description "Enable point-to-multipoint.";
        }
      }
    }
  }
}
container mp2mp {
  description "Configure multipoint-to-multipoint capability.";
  leaf enable {
    type boolean;
    description "Enable multipoint-to-multipoint.";
  }
}

container make-before-break {
  description "Configure make-before-break capability.";
  leaf enable {
    type boolean;
    description "Enable make-before-break.";
  }
}

container hub-and-spoke {
  description "Configure hub-and-spoke-multipoint capability.";
  reference "RFC7140: LDP Extensions for Hub and Spoke Multipoint Label Switched Path";
  leaf enable {
    type boolean;
    description "Enable hub-and-spoke-multipoint.";
  }
}

container node-protection {
  description "Configure node-protection capability.";
  reference "RFC7715: mLDP Node Protection.";
  leaf plr {
    type boolean;
    description "Point of Local Repair capable for MP LSP node protection.";
  }
  leaf merge-point {
    type boolean;
    description "Merge Point capable for MP LSP node protection.";
  }
}

// mldp

} // capability
} // received-peer-state

container session-holdtime {
    description "Session holdtime state.";
    leaf peer {
        type uint16;
        units seconds;
        description "Peer holdtime.";
    }
    leaf negotiated {
        type uint16;
        units seconds;
        description "Negotiated holdtime.";
    }
    leaf remaining {
        type uint16;
        units seconds;
        description "Remaining holdtime.";
    }
} // session-holdtime

leaf session-state {
    type enumeration {
        enum non-existent {
            description "NON EXISTENT state. Transport disconnected.";
        }
        enum initialized {
            description "INITIALIZED state.";
        }
        enum openrec {
            description "OPENREC state.";
        }
        enum opensent {
            description "OPENSENT state.";
        }
        enum operational {
            description "OPERATIONAL state.";
        }
    }
    description
    "Representing the operational status.";
}

container tcp-connection {
    description "TCP connection state.";
    leaf local-address {
        type inet:ip-address;
    }
leaf local-port {
  type inet:port-number;
  description "Local port.";
}
leaf remote-address {
  type inet:ip-address;
  description "Remote address.";
}
leaf remote-port {
  type inet:port-number;
  description "Remote port.";
}

container statistics {
  description "Statistics objects.";
  leaf discontinuity-time {
    type yang:date-and-time;
    mandatory true;
    description "The time on the most recent occasion at which any one or more of this interface’s counters suffered a discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem re-initialized itself.";
  }
  container received {
    description "Inbound statistics.";
    uses statistics-peer-received-sent;
  }
  container sent {
    description "Outbound statistics.";
    uses statistics-peer-received-sent;
  }
  leaf total-addresses {
    type uint32;
  }
}
description
    "The number of learned addresses."
};
leaf total-labels {
  type uint32;
  description
    "The number of learned labels."
};
leaf total-fec-label-bindings {
  type uint32;
  description
    "The number of learned label-address bindings."
};
} // statistics
} // peer-state-derived

grouping policy-container {
  description
    "LDP policy attributes.";
  container label-policy {
    description
      "Label policy attributes.";
    container independent-mode {
      description
        "Independent label policy attributes.";
      container assign {
        if-feature policy-label-assignment-config;
        description
          "Label assignment policies";
        choice prefix-option {
          description
            "Use either prefix-list or host-routes-only.";
        case prefix-list {
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Assign labels according to certain prefixes.";
          }
        }
        case host-routes-only {
          leaf host-routes-only {
            type boolean;
            description
              "'true' to apply host routes only.";
          }
        }
      } // prefix-option
    }
  }
} // policy-container
container advertise {
  description
  "Label advertising policies.";
  container explicit-null {
    description
    "Enables an egress router to advertise an
    explicit null label (value 0) in place of an
    implicit null label (value 3) to the
    penultimate hop router.";
    leaf enable {
      type boolean;
      description
      "'true' to enable explicit null."
    }
    leaf prefix-list {
      type prefix-list-ref;
      description
      "Prefix list name. Applies the filters in the
      specified prefix list to label
      advertisements.
      If the prefix list is not specified, explicit
      null label advertisement is enabled for all
      directly connected prefixes.";
    }
  }
  leaf prefix-list {
    type prefix-list-ref;
    description
    "Applies the prefix list to outgoing label
    advertisements.";
  }
}

container accept {
  description
  "Label advertisement acceptance policies.";
  leaf prefix-list {
    type prefix-list-ref;
    description
    "Applies the prefix list to incoming label
    advertisements.";
  }
}

} // independent-mode
container ordered-mode {
  if-feature policy-ordered-label-config;
  description
"Ordered label policy attributes.");
container egress-lsr {
    description
        "Egress LSR label assignment policies";
    leaf prefix-list {
        type prefix-list-ref;
        description
            "Assign labels according to certain prefixes.";
    }
}
container advertise {
    description
        "Label advertising policies.";
    leaf prefix-list {
        type prefix-list-ref;
        description
            "Applies the prefix list to outgoing label
                advertisements.";
    }
}
container accept {
    description
        "Label advertisement acceptance policies.";
    leaf prefix-list {
        type prefix-list-ref;
        description
            "Applies the prefix list to incoming label
                advertisements.";
    }
}
} // ordered-mode
} // label-policy
} // policy-container

grouping statistics-peer-received-sent {
    description
        "Inbound and outbound statistic counters.";
    leaf total-octets {
        type yang:counter64;
        description
            "The total number of octets sent or received.";
    }
    leaf total-messages {
        type yang:counter64;
        description
            "The number of messages sent or received.";
    }
    leaf address {

type yang:counter64;
  description
    "The number of address messages sent or received.";
}
leaf address-withdraw {
  type yang:counter64;
  description
    "The number of address-withdraw messages sent or received.";
}
leaf initialization {
  type yang:counter64;
  description
    "The number of initialization messages sent or received.";
}
leaf keepalive {
  type yang:counter64;
  description
    "The number of keepalive messages sent or received.";
}
leaf label-abort-request {
  type yang:counter64;
  description
    "The number of label-abort-request messages sent or received.";
}
leaf label-mapping {
  type yang:counter64;
  description
    "The number of label-mapping messages sent or received.";
}
leaf label-release {
  type yang:counter64;
  description
    "The number of label-release messages sent or received.";
}
leaf label-request {
  type yang:counter64;
  description
    "The number of label-request messages sent or received.";
}
leaf label-withdraw {
  type yang:counter64;
  description
    "The number of label-withdraw messages sent or received.";
}
leaf notification {
  type yang:counter64;
  description
"The number of messages sent or received."
}
} // statistics-peer-received-sent

/ * Configuration data nodes */

augment "*/rt:routing/rt:control-plane-protocols" {
  description "LDP augmentation."
}

container mpls-ldp {
  presence "Container for LDP protocol.";
  description
    "Container for LDP protocol."
}

container global {
  description
    "Global attributes for LDP."
  container config {
    description
      "Configuration data."
    uses global-attributes;
  }
  container state {
    config false;
    description
      "Operational state data."
    uses global-attributes;
  }
}

container mldp {
  if-feature mldp;
  description
    "mLDP attributes at per instance level. Defining attributes here does not enable any MP capabilities. MP capabilities need to be explicitly enabled under container capability."
  container config {
    description
      "Configuration data."
    leaf enable {
      type boolean;
      description
        "Enable mLDP."
    }
  }
}
container state {
  config false;
  description
    "Operational state data.";
  leaf enable {
    type boolean;
    description
      "Enable mLDP.";
  }
}

list address-family {
  key "afi";
  description
    "Per-af params.";
  leaf afi {
    type ldp-address-family;
    description
      "Address family type value.";
  }
}

container config {
  description
    "Configuration data.";
  container multicast-only-frr {
    if-feature mldp-mofrr;
    description
      "Multicast only FRR (MoFRR) policy.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Enables MoFRR for the specified access list.";
    }
  } // multicast-only-frr
  container recursive-fec {
    description
      "Recursive FEC policy.";
    leaf prefix-list {
      type prefix-list-ref;
      description
        "Enables recursive FEC for the specified access list.";
    }
  } // recursive-for
}

container state {
  config false;
  description
    "Operational state data.";
  leaf enable {
    type boolean;
    description
      "Enable mLDP.";
  }
}
description
"Operational state data."
container multicast-only-frr {
  if-feature mldp-mofrr;

description
"Multicast only FRR (MoFRR) policy."
leaf prefix-list {
  type prefix-list-ref;
  description
  "Enables MoFRR for the specified access list.";
}
} // multicast-only-frr
container recursive-fec {
  description
  "Recursive FEC policy."
  leaf prefix-list {
    type prefix-list-ref;
    description
    "Enables recursive FEC for the specified access list.";
  }
} // recursive-fec
container ipv4 {
  when ".../../afi = 'ipv4'" {
    description
    "Only for IPv4.";
  }
  description
  "IPv4 state information."
  container roots {
    description
    "IPv4 multicast LSP roots."
    list root {
      key "root-address";
      description
      "List of roots for configured multicast LSPs.";
      leaf root-address {
        type inet:ipv4-address;
        description
        "Root address.";
      }

      leaf is-self {
        type boolean;
        description
        "Is self.";
      }
    }
  }
} // ipv4

"This is the root.”;
}

list reachability {
  key "address interface";
  description
    "A next hop for reachability to root, as a RIB view.");
  leaf address {
    type inet:ipv4-address;
    description
      "The next hop address to reach root.";
  }
  leaf interface {
    type mpls-interface-ref;
    description
      "Interface connecting to next-hop.");
  }
  leaf peer {
    type leafref {
      path
        "../.../.../.../.../peers/peer/"
        + "lsr-id";
    }
    description
      "LDP peer from which this next hop can be reached.");
  }
}
} // list root
} // roots

container bindings {
  description
    "mLDP FEC to label bindings.");

container opaque-type-lspid {
  description
    "The type of opaque value element is the generic LSP identifier";
  reference
    "RFC6388: Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths.";

  list fec-label {
    key
      "root-address lsp-id "
      + "recur-root-address recur-rd”;
    description
      ""
"List of FEC to label bindings."
leaf root-address {
    type inet:ipv4-address;
    description "Root address.";
}
leaf lsp-id {
    type uint32;
    description "ID to identify the LSP.";
}
leaf recur-root-address {
    type inet:ipv4-address;
    description "Recursive root address.";
    reference
        "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
leaf recur-rd {
    type route-distinguisher;
    description "Route Distinguisher in the VPN-Recursive Opaque Value.";
    reference
        "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-lspid

container opaque-type-src {
    description
        "The type of opaque value element is the transit source TLV";
    reference
        "RFC6826: Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths.";
    list fec-label {
        key
            "root-address source-address group-address " + "rd recur-root-address recur-rd";
        description
            "List of FEC to label bindings.";
        leaf root-address {
            type inet:ipv4-address;
description
   "Root address.";
}
leaf source-address {
    type inet:ip-address;
    description
       "Source address.";
}
leaf group-address {
    type inet:ip-address-no-zone;
    description
       "Group address.";
}
leaf rd {
    type route-distinguisher;
    description
       "Route Distinguisher.";
    reference
       "RFC7246: Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context.";
}
leaf recur-root-address {
    type inet:ip-address;
    description
       "Recursive root address.";
    reference
       "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
leaf recur-rd {
    type route-distinguisher;
    description
       "Route Distinguisher in the VPN-Recursive Opaque Value.";
    reference
       "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-src

container opaque-type-bidir {
    description
       "The type of opaque value element is the generic LSP identifier";
list fec-label {
  key "root-address rp group-address"
    + "rd recur-root-address recur-rd";
  description "List of FEC to label bindings.";
  leaf root-address {
    type inet:ipv4-address;
    description "Root address.";
  }
  leaf rp {
    type inet:ip-address;
    description "RP address.";
  }
  leaf group-address {
    type inet:ip-address-no-zone;
    description "Group address.";
  }
  leaf rd {
    type route-distinguisher;
    description "Route Distinguisher.";
    reference "RFC7246: Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context.";
  }
  leaf recur-root-address {
    type inet:ip-address;
    description "Recursive root address.";
    reference "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
  }
  leaf recur-rd {
    type route-distinguisher;
    description "Route Distinguisher in the VPN-Recursive"
Opaque Value.

reference
"RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-bidir
} // bindings
} // ipv4

c ontainer ipv6 {
when ".//afi = 'ipv6'" {
  description
  "Only for IPv6.";
} // ipv6

description
"IPv6 state information.";
container roots {
  description
  "IPv6 multicast LSP roots.";
  list root {
    key "root-address";
    description
    "List of roots for configured multicast LSPs.";
    leaf root-address {
      type inet:ipv6-address;
      description
      "Root address.";
    }
  }
  leaf is-self {
    type boolean;
    description
    "This is the root.";
  }
  list reachability {
    key "address interface";
    description
    "A next hop for reachability to root, as a RIB view.";
    leaf address {
      type inet:ipv6-address;
      description
      "The next hop address to reach root.";
    }
  }
} // ipv6
leaf interface {
    type mpls-interface-ref;
    description
    "Interface connecting to next-hop."
}

leaf peer {
    type leafref {
        path
        "../../../../../../../peers/peer/" + "lsr-id";
    }
    description
    "LDP peer from which this next hop can be reached."
}

} // list root
} // roots

container bindings {
    description
    "mLDP FEC to label bindings."
}

container opaque-type-lspid {
    description
    "The type of opaque value element is the generic LSP identifier"
    reference
    list fec-label {
        key
        "root-address lsp-id " + "recur-root-address recur-rd";
        description
        "List of FEC to label bindings."
        leaf root-address {
            type inet:ipv6-address;
            description
            "Root address."
        }
        leaf lsp-id {
            type uint32;
            description "ID to identify the LSP."
        }
        leaf recur-root-address {
            type inet:ip-address;
            description
        }
    }

    } // bindings

"Recursive root address.";
reference
"RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}

leaf recur-rd {
    type route-distinguisher;
description
    "Route Distinguisher in the VPN-Recursive Opaque Value.";
reference
"RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}
uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-lspid

container opaque-type-src {
    description
    "The type of opaque value element is the transit Source TLV";
reference
"RFC6826: Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths.";
list fec-label {
    key
    "root-address source-address group-address " + "rd recur-root-address recur-rd";
description
    "List of FEC to label bindings.";
leaf root-address {
    type inet:ipv6-address;
description
    "Root address.";
}
leaf source-address {
    type inet:ip-address;
description
    "Source address.";
}
leaf group-address {
    type inet:ip-address-no-zone;
description
    "Group address.";
}
leaf rd {
  type route-distinguisher;
  description
    "Route Distinguisher.";
  reference
    "RFC7246: Multipoint Label Distribution
     Protocol In-Band Signaling in a Virtual
     Routing and Forwarding (VRF) Table
     Context.";
}

leaf recur-root-address {
  type inet:ip-address;
  description
    "Recursive root address.";
  reference
    "RFC6512: Using Multipoint LDP When the
     Backbone Has No Route to the Root";
}

leaf recur-rd {
  type route-distinguisher;
  description
    "Route Distinguisher in the VPN-Recursive
     Opaque Value.";
  reference
    "RFC6512: Using Multipoint LDP When the
     Backbone Has No Route to the Root";
  uses mldp-binding-label-state-attributes;
} // fec-label
} // opaque-type-src

container opaque-type-bidir {
  description
    "The type of opaque value element is
     the generic LSP identifier";
  reference
    "RFC6826: Multipoint LDP In-Band Signaling for
     Point-to-Multipoint and
     Multipoint-to-Multipoint Label Switched
     Paths.";
  list fec-label {
    key
      "root-address rp group-address " + "rd recur-root-address recur-rd";
    description
      "List of FEC to label bindings.";
    leaf root-address {
      type inet:ipv6-address;
    }
  } // fec-label
} // opaque-type-src
leaf rp {
  type inet:ip-address;
  description
    "RP address.";
}

leaf group-address {
  type inet:ip-address-no-zone;
  description
    "Group address.";
}

leaf rd {
  type route-distinguisher;
  description
    "Route Distinguisher.";
  reference
    "RFC7246: Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context.";
}

leaf recur-root-address {
  type inet:ip-address;
  description
    "Recursive root address.";
  reference
    "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}

leaf recur-rd {
  type route-distinguisher;
  description
    "Route Distinguisher in the VPN-Recursive Opaque Value.";
  reference
    "RFC6512: Using Multipoint LDP When the Backbone Has No Route to the Root";
}

uses mldp-binding-label-state-attributes;

container configured-leaf-lsps {
description
"Configured multicast LSPs."

container p2mp {
  description
  "Configured point-to-multipoint LSPs."
  uses mldp-configured-lsp-roots;
}
container mp2mp {
  description
  "Configured multipoint-to-multipoint LSPs."
  uses mldp-configured-lsp-roots;
}
} // configured-leaf-lsps
} // list address-family
} // mldp

list address-family {
  key "afi";
  description
  "Per-vrf per-af params.";
  leaf afi {
    type ldp-address-family;
    description
    "Address family type value.";
  }
}

container config {
  description
  "Configuration data.";
  leaf enable {
    type boolean;
    description
    "'true' to enable the address family.";
  }
  uses policy-container;
}

container ipv4 {
  when "../af = ipv4" {
    description
    "Only for IPv4.";
  }
  description
  "IPv4 address family.";
  leaf transport-address {
    type inet:ipv4-address;
    description
    "The transport address advertised in LDP Hello
messages.
}
} // ipv4
container ipv6 {
  when "../../afi = 'ipv6'" {
    description
    "Only for IPv6."
  }
  description
  "IPv6 address family.";
  leaf transport-address {
    type inet:ipv6-address;
    description
    "The transport address advertised in LDP Hello messages."
  }
} // ipv6
}
  container state {
    config false;
    description
    "Operational state data."
    leaf enable {
      type boolean;
      description
      "'true' to enable the address family."
    }
  }
uses policy-container;

container ipv4 {
  when "../../afi = 'ipv4'" {
    description
    "Only for IPv4."
  }
  description
  "IPv4 address family.";
  leaf transport-address {
    type inet:ipv4-address;
    description
    "The transport address advertised in LDP Hello messages."
  }
}

container bindings {
  description
  "LDP address and label binding information.";
  list address {
    ...
key "address";
description
    "List of address bindings.";
leaf address {
    type inet:ipv4-address;
description
    "Binding address.";
}
uses binding-address-state-attributes;
} // binding-address

list fec-label {
    key "fec";
description
    "List of label bindings.";
leaf fec {
    type inet:ipv4-prefix;
description
    "Prefix FEC.";
}
uses binding-label-state-attributes;
} // fec-label
} // binding
} // ipv4

container ipv6 {
    when "/..//afi = 'ipv6'" {
        description
        "Only for IPv6.";
    }
description
    "IPv6 address family."
leaf transport-address {
    type inet:ipv6-address;
description
    "The transport address advertised in LDP Hello messages.";
}
}

container binding {
    description
    "LDP address and label binding information.";
list address {
    key "address";
description
    "List of address bindings.";
leaf address {
    type inet:ipv6-address;
description
    "List of address bindings.";
leaf address {
    type inet:ipv6-address;
description
"Binding address."
)
uses binding-address-state-attributes;
) // binding-address

list fec-label {
    key "fec";
description
    "List of label bindings.";
leaf fec {
    type inet:ipv6-prefix;
description
    "Prefix FEC.";
}
uses binding-label-state-attributes;
) // fec-label
) // binding
) // ipv6
) // state
} // address-family

container discovery {
description
    "Neighbor discovery configuration.";
}

container interfaces {
description
    "A list of interfaces for basic discovery.";
container config {
description
    "Configuration data.";
    uses basic-discovery-timers;
}
container state {
    config false;
    description
    "Operational state data.";
    uses basic-discovery-timers;
}

list interface {
    key "interface";
description
    "List of LDP interfaces.";
leaf interface {
    type mpls-interface-ref;
description
    }
"Interface."
}
}

container config {
  description "Configuration data.";
  uses basic-discovery-timers {
    if-feature per-interface-timer-config;
  }
  leaf igp-synchronization-delay {
    if-feature per-interface-timer-config;
    type uint16 {
      range 3..60;
    }
    units seconds;
    description "Sets the interval that the LDP waits before notifying the Interior Gateway Protocol (IGP) that label exchange is completed so that IGP can start advertising the normal metric for the link.";
  }
}

container state {
  config false;
  description "Operational state data.";
  uses basic-discovery-timers {
    if-feature per-interface-timer-config;
  }
  leaf igp-synchronization-delay {
    if-feature per-interface-timer-config;
    type uint16 {
      range 3..60;
    }
    units seconds;
    description "Sets the interval that the LDP waits before notifying the Interior Gateway Protocol (IGP) that label exchange is completed so that IGP can start advertising the normal metric for the link.";
  }
  leaf next-hello {
    type uint16;
    units seconds;
    description "Time to send the next hello message.";
  }
}
} // state
list address-family {
  key "afi";
  description "Per-vrf per-af params.";
  leaf afi {
    type ldp-address-family;
    description "Address family type value.";
  }
  container config {
    description "Configuration data.";
    leaf enable {
      type boolean;
      description "Enable the address family on the interface.";
    }
  }
  container ipv4 {
    must "/if:interfaces/if:interface" + "[name = current()//interface]/" + "ip:ipv4" {
      description "Only if IPv4 is enabled on the interface.";
    }
    description "IPv4 address family.";
    leaf transport-address {
      type union {
        type enumeration {
          enum "use-interface-address" {
            description "Use interface address as the transport address.";
          }
        }
      }
      type inet:ipv4-address;
    }
    description "IP address to be advertised as the LDP transport address.";
  }
  }
  container ipv6 {
    must "/if:interfaces/if:interface" + "[name = current()//interface]/" + "ip:ipv6" {
      description "IPv6 address family.";
      leaf transport-address {
        type union {
          type enumeration {
            enum "use-interface-address" {
              description "Use interface address as the transport address.";
            }
          }
        }
      }
      type inet:ipv6-address;
    }
    description "IP address to be advertised as the LDP transport address.";
  }
}
description
"Only if IPv6 is enabled on the interface."
}
description
"IPv6 address family."
leaf transport-address {
type union {
type enumeration {
  enum "use-interface-address" {
    description
    "Use interface address as the transport
    address."
  }
}
type inet:ipv4-address;
}
description
"IP address to be advertised as the LDP transport address."
}
} // ipv6
}
container state {
config false;
description
"Operational state data."
leaf enable {
type boolean;
description
"Enable the address family on the interface."
}
container ipv4 {
must "/if:interfaces/if:interface"
+ "[name = current()//..//..//interface]"
+ "ip:ipv4" {
  description
  "Only if IPv4 is enabled on the interface."
}
description
"IPv4 address family."
leaf transport-address {
type union {
type enumeration {
  enum "use-interface-address" {
    description
    "Use interface address as the transport
address.
}

}  

type inet:ipv4-address;
}
description
"IP address to be advertised as the LDP transport address.";
}

list hello-adjacencies {
  key "adjacent-address";
  description "List of hello adjacencies.";

  leaf adjacent-address {
    type inet:ipv4-address;
    description
    "Neighbor address of the hello adjacency.";
  }
}

uses adjacency-state-attributes;

leaf peer {
  type leafref {
    path ".//.../.../.../.../.../.../peers/peer/"
    + "lsr-id";
  }
  description
  "LDP peer from this adjacency.";
}
}
}  // hello-adjacencies

container ipv6 {
  must "/if:interfaces/if:interface"
  + "[name = current()//...//../../interface]/"
  + "ip:ipv6" {
    description
    "Only if IPv6 is enabled on the interface.";
  }
  description
  "IPv6 address family.";
  leaf transport-address {
    type union {
      type enumeration {
        enum "use-interface-address" {
          description
          "Use interface address as the transport address.";
        }
      }
    }
  }
}

type inet:ipv4-address;
}

description
"IP address to be advertised as the LDP transport address.";
}

list hello-adjacencies {
  key "adjacent-address";
  description "List of hello adjacencies.";

  leaf adjacent-address {
    type inet:ipv6-address;
    description
    "Neighbor address of the hello adjacency.";
  }
  uses adjacency-state-attributes;

  leaf peer {
    type leafref {
      path "../../../../../../../peers/peer/"
          + "lsr-id";
    }
    description
    "LDP peer from this adjacency.";
  }
} // hello-adjacencies
} // ipv6
} // address-family
} // list interface
} // interfaces

container targeted {
  description
  "A list of targeted neighbors for extended discovery.";
  container config {
    description
    "Configuration data.";
    uses extended-discovery-timers;
    uses extended-discovery-policy-attributes;
  }
  container state {
config false;
description
  "Operational state data.";
uses extended-discovery-timers;
uses extended-discovery-policy-attributes;
}

list address-family {
  key "afi";
  description
    "Per-af params.";
  leaf afi {
    type ldp-address-family;
    description
      "Address family type value.";
  }
}

count state {
  config false;
  description
    "Operational state data.";
}

count ipv4 {
  when ":/afi = 'ipv4'" {
    description
      "For IPv4.";
  }
  description
    "IPv4 address family.";
}

list hello-adjacencies {
  key "local-address adjacent-address";
  description "List of hello adjacencies.";
}

  leaf local-address {
    type inet:ipv4-address;
    description
      "Local address of the hello adjacency.";
  }
  leaf adjacent-address {
    type inet:ipv4-address;
    description
      "Neighbor address of the hello adjacency.";
  }
}

uses adjacency-state-attributes;

leaf peer {
  type leafref {

path "./././././././././././peers/peer/"
  + "lsr-id";
}
description
  "LDP peer from this adjacency."
}
} // hello-adjacencies
} // ipv4

container ipv6 {
  when "././afi = 'ipv6'" {
    description
      "For IPv6.";
  }
  description
    "IPv6 address family.";
  list hello-adjacencies {
    key "local-address adjacent-address";
    description "List of hello adjacencies.";
    leaf local-address {
      type inet:ipv6-address;
      description
        "Local address of the hello adjacency.";
    }
    leaf adjacent-address {
      type inet:ipv6-address;
      description
        "Neighbor address of the hello adjacency.";
    }
  }
  uses adjacency-state-attributes;

  leaf peer {
    type leafref {
      path "./././././././././././peers/peer/"
        + "lsr-id";
    }
    description
      "LDP peer from this adjacency.";
  }
} // hello-adjacencies
} // ipv6
} // state

container ipv4 {
  when "./afi = 'ipv4'" {
    description
      "IPv4 address family.";
    list hello-adjacencies {
      key "local-address adjacent-address";
      description "List of hello adjacencies.";
      leaf local-address {
        type inet:ipv4-address;
        description
          "Local address of the hello adjacency.";
      }
      leaf adjacent-address {
        type inet:ipv4-address;
        description
          "Neighbor address of the hello adjacency.";
      }
    }
    uses adjacency-state-attributes;

    leaf peer {
      type leafref {
        path "./././././././././././peers/peer/"
          + "lsr-id";
      }
      description
        "LDP peer from this adjacency.";
    }
  }
} // hello-adjacencies
} // ipv4
} // state
"For IPv4."
}

description
"IPv4 address family.";
list target {
    key "adjacent-address";
    description
    "Targeted discovery params.";

    leaf adjacent-address {
        type inet:ipv4-address;
        description
        "Configures a remote LDP neighbor and enables
        extended LDP discovery of the specified
        neighbor.";
    }

    container config {
        description
        "Configuration data.";
        leaf enable {
            type boolean;
            description
            "Enable the target.";
        }

        leaf local-address {
            type inet:ipv4-address;
            description
            "The local address.";
        }
    }

    container state {
        config false;
        description
        "Operational state data.";
        leaf enable {
            type boolean;
            description
            "Enable the target.";
        }

        leaf local-address {
            type inet:ipv4-address;
            description
            "The local address.";
        }
    }
} // state
}

// ipv6
container ipv6 {

when "./afi = 'ipv6'" {
    description
    "For IPv6.";
}
description
"IPv6 address family."
list target {
    key "adjacent-address";
    description
    "Targeted discovery params.";
    leaf adjacent-address {
        type inet:ipv6-address;
        description
        "Configures a remote LDP neighbor and enables
        extended LDP discovery of the specified
        neighbor.";
    }
    container config {
        description
        "Configuration data.";
        leaf enable {
            type boolean;
            description
            "Enable the target.";
        }
        leaf local-address {
            type inet:ipv6-address;
            description
            "The local address.";
        }
    }
    container state {
        config false;
        description
        "Operational state data.";
        leaf enable {
            type boolean;
            description
            "Enable the target.";
        }
        leaf local-address {
            type inet:ipv6-address;
            description
            "The local address.";
        }
    } // state
}
container forwarding-nexthop {
  if-feature forwarding-nexthop-config;
  description
  "Configuration for forwarding nexthop.";
}

container interfaces {
  description
  "A list of interfaces on which forwarding is disabled.";
  list interface {
    key "interface";
    description
    "List of LDP interfaces.";
    leaf interface {
      type mpls-interface-ref;
      description
      "Interface.";
    }
  }
  list address-family {
    key "afi";
    description
    "Per-vrf per-af params.";
    leafafi {
      type ldp-address-family;
      description
      "Address family type value.";
    }
  }
  container config {
    description
    "Configuration data.";
    leaf ldp-disable {
      type boolean;
      description
      "Disable LDP forwarding on the interface.";
    }
    leaf mldp-disable {
      if-feature mldp;
      type boolean;
      description
      "Disable mLDP forwarding on the interface.";
    }
  }
}
container state {
  config false;
  description "Operational state data.";
  leaf ldp-disable {
    type boolean;
    description "Disable LDP forwarding on the interface.";
  }
  leaf mldp-disable {
    if-feature mldp;
    type boolean;
    description "Disable mLDP forwarding on the interface.";
  }
}
} // address-family
} // list interface
} // interfaces
} // forwarding-nexthop
uses policy-container {
  if-feature all-af-policy-config;
}
} // global

container peers {
  description "Peers configuration attributes.";

  container config {
    description "Configuration data.";
    uses peer-authentication {
      if-feature global-session-authentication;
    }
    uses peer-attributes;

    container session-downstream-on-demand {
      if-feature session-downstream-on-demand-config;
      description "Session downstream-on-demand attributes.";
      leaf enable {
        type boolean;
        description "'true' if session downstream-on-demand is enabled.";
      }
    }
    leaf peer-list {
type peer-list-ref;
description
   "The name of a peer ACL.";
}
}
}

container state {
  config false;
description
   "Operational state data.";
  uses peer-authentication {
    if-feature global-session-authentication;
  }
  uses peer-attributes;

carrier session-downstream-on-demand {
  if-feature session-downstream-on-demand-config;
description
   "Session downstream-on-demand attributes.";
  leaf enable {
    type boolean;
description
       "'true' if session downstream-on-demand is enabled.";
  }
  leaf peer-list {
    type peer-list-ref;
description
       "The name of a peer ACL.";
  }
}

list peer {
  key "lsr-id";
description
   "List of peers.";

  leaf lsr-id {
    type yang:dotted-quad;
description
       "LSR ID.";
  }

carrier config {
  description
     "Configuration data.";
  leaf admin-down {
    type boolean;
default false;
  }
}
description
    "'true' to disable the peer.";
}

container capability {
    description
        "Per peer capability";
    container mldp {
        if-feature mldp;
        description
            "mLDP capabilities.";
        uses mldp-capabilities;
    }
}

uses peer-af-policy-container {
    if-feature all-af-policy-config;
}

uses peer-authentication;

uses graceful-restart-attributes-per-peer {
    if-feature per-peer-graceful-restart-config;
}

uses peer-attributes {
    if-feature per-peer-session-attributes-config;
}

container address-family {
    description
        "Per-vrf per-af params.";
    container ipv4 {
        description
            "IPv4 address family.";
        uses peer-af-policy-container;
    }
    container ipv6 {
        description
            "IPv6 address family.";
        uses peer-af-policy-container;
    }
    // ipv6
    // address-family
}

container state {
    config false;
    description
        "Operational state data.";
leaf admin-down {
    type boolean;
    default false;
    description
        "'true' to disable the peer."
}

container capability {
    description
        "Per peer capability";
    container mldp {
        if-feature mldp;
        description
            "mLDP capabilities.";
        uses mldp-capabilities;
    }
}

uses peer-af-policy-container {
    if-feature all-af-policy-config;
}

uses peer-authentication;

uses graceful-restart-attributes-per-peer {
    if-feature per-peer-graceful-restart-config;
}

uses peer-attributes {
    if-feature per-peer-session-attributes-config;
}

container address-family {
    description
        "Per-vrf per-af params.";
    container ipv4 {
        description
            "IPv4 address family.";
        uses peer-af-policy-container;
        list hello-adjacencies {
            key "local-address adjacent-address";
            description "List of hello adjacencies.";
            leaf local-address {
                type inet:ipv4-address;
                description
                    "Local address of the hello adjacency.";
            }
        }
    }
}

rpc mpls-ldp-clear-peer {
  description "Clears the session to the peer.";
  input {
    leaf lsr-id {
      type union {
        type yang:dotted-quad;
        type uint32;
      } description "LSR ID of peer to be cleared. If this is not provided then all peers are cleared";
    }
  }
}

rpc mpls-ldp-clear-hello-adjacency {
  description "Clears the hello adjacency";
  input {
    container hello-adjacency {
      description "Link adjacency or targeted adjacency. If this is not provided then all hello adjacencies are cleared";
      choice hello-adjacency-type {
        description "Adjacency type.";
        case targeted {
          container targeted {
            presence "Present to clear targeted adjacencies.";
            description "Clear targeted adjacencies.";
            leaf target-address {
              type inet:ip-address;
              description "The target address. If this is not provided then all targeted adjacencies are cleared";
            }
          }
        }
        case link {
          container link {
          }
        }
      }
    }
  }
}
presence "Present to clear link adjacencies."
  description
  "Clear link adjacencies."
leaf next-hop-interface {
  type mpls-interface-ref;
  description
  "Interface connecting to next-hop. If this is not provided then all link adjacencies are cleared."
}
leaf next-hop-address {
  type inet:ip-address;
  must "../next-hop-interface" {
    description
    "Applicable when interface is specified."
  }
  description
  "IP address of next-hop. If this is not provided then adjacencies to all next-hops on the given interface are cleared."
} // next-hop-address
} // link
}
}
}

crpc mpls-ldp-clear-peer-statistics {
  description
  "Clears protocol statistics (e.g. sent and received counters)."
  input {
    leaf lsr-id {
      type union {
        type yang:dotted-quad;
        type uint32;
      }
      description
      "LSR ID of peer whose statistic are to be cleared. If this is not provided then all peers statistics are cleared"
    }
  }
}
/*
* Notifications
notification mpls-ldp-peer-event {
    description "Notification event for a change of LDP peer operational status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-peer-ref;
}

notification mpls-ldp-hello-adjacency-event {
    description "Notification event for a change of LDP adjacency operational status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-adjacency-ref;
}

notification mpls-ldp-fec-event {
    description "Notification event for a change of FEC status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses ldp-fec-event;
}

notification mpls-mldp-fec-event {
    description "Notification event for a change of FEC status.";
    leaf event-type {
        type oper-status-event-type;
        description "Event type.";
    }
    uses mldp-fec-event;
}

<CODE ENDS>
7. Security Considerations

The configuration, state, action and notification data defined using YANG data models in this document are likely to be accessed via the protocols such as NETCONF [RFC6241] etc.

Hence, YANG implementations MUST comply with the security requirements specified in section 15 of [RFC6020]. Additionally, NETCONF implementations MUST comply with the security requirements specified in sections 2.2, 2.3 and 9 of [RFC6241] as well as section 3.7 of [RFC6536].

8. IANA Considerations

This document does not extend LDP or mLDP base protocol specification and hence there are no IANA considerations.

Note to the RFC Editor: Please remove IANA section before the publication.

9. Acknowledgments

The authors would like to acknowledge Eddie Chami, Nagendra Kumar, Mannan Venkatesan, Pavan Beeram for their contribution to this document. We also acknowledge Ladislav Lhotka for his useful comments as the YANG Doctor.

10. References

10.1. Normative References

[I-D.ietf-netmod-routing-cfg]

[I-D.rtgyangdt-rtgw-ni-model]

[I-D.saad-mpls-base-yang]


10.2. Informative References


Appendix A. Additional Contributors

Stephane Litkowski
Orange.
Email: stephane.litkowski@orange.com

Reshad Rahman
Cisco Systems Inc.
Email: rrahman@cisco.com
Abstract

This document defines a YANG data model for the management of Path Computation Element communications Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs. The data model includes configuration data and state data (status information and counters for the collection of statistics).

Status of This Memo

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

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

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

This Internet-Draft will expire on 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.
The Path Computation Element (PCE) defined in [RFC4655] is an entity that is capable of computing a network path or route based on a network graph, and applying computational constraints. A Path
Computation Client (PCC) may make requests to a PCE for paths to be computed.

PCEP is the communication protocol between a PCC and PCE and is defined in [RFC5440]. PCEP interactions include path computation requests and path computation replies as well as notifications of specific states related to the use of a PCE in the context of Multiprotocol Label Switching (MPLS) and Generalized MPLS (GMPLS) Traffic Engineering (TE). [I-D.ietf-pce-stateful-pce] specifies extensions to PCEP to enable stateful control of MPLS TE LSPs.

This document defines a YANG [RFC6020] data model for the management of PCEP speakers. It is important to establish a common data model for how PCEP speakers are identified, configured, and monitored. The data model includes configuration data and state data (status information and counters for the collection of statistics).

This document contains a specification of the PCEP YANG module, "ietf-pcep" which provides the PCEP [RFC5440] data model.

2. Requirements Language

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

3. Terminology and Notation

This document uses the terminology defined in [RFC4655] and [RFC5440]. In particular, it uses the following acronyms.

- Path Computation Request message (PCReq).
- Path Computation Reply message (PCRep).
- Notification message (PCNtf).
- Error message (PCErr).
- Request Parameters object (RP).
- Synchronization Vector object (SVEC).
- Explicit Route object (ERO).

This document also uses the following terms defined in [RFC7420]:

- PCEP entity: a local PCEP speaker.
o PCEP peer: to refer to a remote PCEP speaker.

o PCEP speaker: where it is not necessary to distinguish between local and remote.

Further, this document also uses the following terms defined in [I-D.ietf-pce-stateful-pce]:

o Stateful PCE, Passive Stateful PCE, Active Stateful PCE

o Delegation, Revocation, Redelegation

o LSP State Report, Path Computation Report message (PCRpt).

o LSP State Update, Path Computation Update message (PCUpd).

[I-D.ietf-pce-pce-initiated-lsp]:

o PCE-initiated LSP, Path Computation LSP Initiate Message (PCInitiate).

[I-D.ietf-pce-lsp-setup-type]:

o Path Setup Type (PST).

[I-D.ietf-pce-segment-routing]:

o Segment Routing (SR).

3.1. Tree Diagrams

A graphical representation of the complete data tree is presented in Section 5. The meaning of the symbols in these diagrams is as follows and as per [I-D.ietf-netmod-rfc6087bis]:

o Brackets "[" and "]" enclose list keys.

o Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).

o Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

o Ellipsis ("...") stands for contents of subtrees that are not shown.
3.2. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>key-chain</td>
<td>ietf-key-chain</td>
<td>[I-D.ietf-rtgwg-yang-key-chain]</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

4. Objectives

This section describes some of the design objectives for the model:

- In case of existing implementations, it needs to map the data model defined in this document to their proprietary native data model. To facilitate such mappings, the data model should be simple.

- The data model should be suitable for new implementations to use as is.

- Mapping to the PCEP MIB Module should be clear.

- The data model should allow for static configurations of peers.

- The data model should include read-only counters in order to gather statistics for sent and received PCEP messages, received messages with errors, and messages that could not be sent due to errors.

- It should be fairly straightforward to augment the base data model for advanced PCE features.
5. The Design of PCEP Data Model

The module, "ietf-pcep", defines the basic components of a PCE speaker.

module: ietf-pcep
  +--rw pcep
     |  +--rw entity
     |     +--rw addr inet:ip-address
     |     +--rw enabled? boolean
     |     +--rw role pcep-role
     |     +--rw description? string
     |     +--rw speaker-entity-id? string {stateful-sync-opt}?
     |  +--rw domain
     |     +--rw domain* [domain-type domain]
     |     +--rw domain-type domain-type
     |     +--rw domain domain
     +--rw capability
     |  +--rw gmpls? boolean {gmpls}?
     |  +--rw bi-dir? boolean
     |  +--rw diverse? boolean
     |  +--rw load-balance? boolean
     |  +--rw synchronize? boolean {svec}?
     |  +--rw objective-function? boolean {objective-function}?
     |  +--rw add-path-constraint? boolean
     |  +--rw prioritization? boolean
     |  +--rw multi-request? boolean
     |  +--rw gco? boolean {gco}?
     |  +--rw p2mp? boolean {p2mp}?
     +--rw stateful {stateful}?
     |  +--rw enabled? boolean
     |  +--rw active? boolean
     |  +--rw pce-initiated? boolean {pce-initiated}?
     |  +--rw include-db-ver? boolean {stateful-sync-opt}?
     |  +--rw trigger-resync? boolean {stateful-sync-opt}?
     |  +--rw trigger-initial-sync? boolean {stateful-sync-opt}?
     |  +--rw incremental-sync? boolean {stateful-sync-opt}?
     |  +--rw sr {sr}?
     |     +--rw enabled? boolean
     +--rw pce-info
     +--rw scope
     |  +--rw intra-area-scope? boolean
     |  +--rw intra-area-pref? uint8
     |  +--rw inter-area-scope? boolean
     |  +--rw inter-area-scope-default? boolean
     |  +--rw inter-as-scope? boolean
     |  +--rw inter-as-scope-default? boolean
---rw dead-timer?          uint32
---rw allow-negotiation?  boolean
---rw max-keep-alive-timer? uint32
---rw max-dead-timer?      uint32
---rw min-keep-alive-timer? uint32
---rw min-dead-timer?      uint32
---rw sync-timer?          uint32 {svec}? 
---rw request-timer?       uint32
---rw max-sessions?        uint32
---rw max-unknown-reqs?    uint32
---rw max-unknown-msgs?    uint32
---rw pcep-notification-max-rate uint32
---rw stateful-parameter {stateful}? 
   +--rw state-timeout?      uint32 
   +--rw redelegation-timeout? uint32
   +--rw rpt-non-pcep-lsp?   boolean
---rw of-list {objective-function}? 
   +--rw objective-function* [of] 
   +--rw of objective-function
---rw peers 
   +--rw peer* [addr] 
      +--rw addr     inet:ip-address
      +--rw description?  string
   +--rw domain 
      +--rw domain* [domain-type domain] 
      +--rw domain-type  domain-type
      +--rw domain  domain
---rw capability 
   +--rw gmpls?           boolean {gmpls}? 
   +--rw bi-dir?          boolean
   +--rw diverse?         boolean
   +--rw load-balance?    boolean
   +--rw synchronize?     boolean {svec}?  
   +--rw objective-function?  boolean {objective-function}? 
   +--rw add-path-constraint? boolean 
   +--rw prioritization?  boolean
   +--rw multi-request?   boolean
   +--rw gco?             boolean {gco}? 
   +--rw p2mp?           boolean {p2mp}? 
---rw stateful {stateful}? 
   +--rw enabled?          boolean
   +--rw active?           boolean
   +--rw pce-initiated?    boolean {pce-initiated}? 
   +--rw include-db-ver?   boolean {stateful-sync-opt}? 
   +--rw trigger-resync?   boolean {stateful-sync-opt}? 
   +--rw trigger-initial-sync? boolean {stateful-sync-opt}? 
   +--rw incremental-sync? boolean {stateful-sync-opt}? 
---rw sr {sr}?
---rw enabled?  boolean

---rw scope

   ---rw intra-area-scope?  boolean
   ---rw intra-area-pref?  uint8
   ---rw inter-area-scope?  boolean
   ---rw inter-area-scope-default?  boolean
   ---rw inter-area-pref?  uint8
   ---rw inter-as-scope?  boolean
   ---rw inter-as-scope-default?  boolean
   ---rw inter-as-pref?  uint8
   ---rw inter-layer-scope?  boolean
   ---rw inter-layer-pref?  uint8

---rw neigh-domains

   ---rw domain* [domain-type domain]
       ---rw domain-type  domain-type
       ---rw domain  domain

---rw delegation-pref?  uint8  {stateful}?  

---rw (auth-type-selection)?

     +--:(auth-key-chain)
     |  ---rw key-chain?  key-chain: key-chain-ref
     +--:(auth-key)
     |  ---rw key?  string
     +--rw crypto-algorithm
          +--rw (algorithm)?
          |     +--:(hmac-sha-1-12)  {crypto-hmac-sha-1-12}?
          |         |  ---rw hmac-sha-1-12?  empty
          |     +--:(aes-cmac-prf-128)  {aes-cmac-prf-128}?
          |         |  ---rw aes-cmac-prf-128?  empty
          |     +--:(md5)
          |         |  ---rw md5?  empty
          |     +--:(sha-1)
          |         |  ---rw sha-1?  empty
          |     +--:(hmac-sha-1)
          |         |  ---rw hmac-sha-1?  empty
          |     +--:(hmac-sha-256)
          |         |  ---rw hmac-sha-256?  empty
          |     +--:(hmac-sha-384)
          |         |  ---rw hmac-sha-384?  empty
          |     +--:(hmac-sha-512)
          |         |  ---rw hmac-sha-512?  empty
          |     +--:(clear-text)  {clear-text}?
          |         |  ---rw clear-text?  empty
          +--:(replay-protection-only)  {replay-protection-only}

     +--rw replay-protection-only?  empty
     +--:(auth-tls)  {tls}?
     +--rw tls

---rw of-list  {objective-function}?

     +--rw objective-function*  [of]
```yang
  ++--ro pcep-state
     ++--ro entity
        ++--ro addr?               inet:ip-address
        ++--ro index?              uint32
        ++--ro admin-status?       pcep-admin-status
        ++--ro oper-status?        pcep-admin-status
        ++--ro role?               pcep-role
        ++--ro description?        string
        ++--ro speaker-entity-id?  string {stateful-sync-opt}?
  ++--ro domain
     ++--ro domain* [domain-type domain]
     ++--ro domain-type        domain-type
     ++--ro domain            domain
  ++--ro capability
     ++--ro gmpls?             boolean {gmpls}?
     ++--ro bi-dir?            boolean
     ++--ro diverse?           boolean
     ++--ro load-balance?      boolean
     ++--ro synchronize?       boolean {svec}?
     ++--ro objective-function? boolean {objective-function}?
     ++--ro add-path-constraint? boolean
     ++--ro prioritization?    boolean
     ++--ro multi-request?     boolean
     ++--ro gco?               boolean {gco}?
     ++--ro p2mp?              boolean {p2mp}?
     ++--ro stateful {stateful}?
        ++--ro enabled?           boolean
        ++--ro active?            boolean
        ++--ro pce-initiated?     boolean {pce-initiated}?
        ++--ro include-db-ver?    boolean {stateful-sync-opt}?
        ++--ro trigger-resync?    boolean {stateful-sync-opt}?
        ++--ro trigger-initial-sync? boolean {stateful-sync-opt}?
        ++--ro incremental-sync?  boolean {stateful-sync-opt}?
        ++--ro sr {sr}?
        ++--ro enabled?           boolean
  ++--ro pce-info
     ++--ro scope
        ++--ro intra-area-scope?  boolean
        ++--ro intra-area-pref?   uint8
        ++--ro inter-area-scope?  boolean
        ++--ro inter-area-scope-default? boolean
        ++--ro inter-area-pref?   uint8
        ++--ro inter-as?          boolean
        ++--ro inter-as-scope?    boolean
        ++--ro inter-as-scope-default? boolean
        ++--ro inter-as-pref?     uint8
        ++--ro inter-layer-scope? boolean
        ++--ro inter-layer-pref?  uint8
```

++-ro neigh-domains
  ++-ro domain* [domain-type domain]
    ++-ro domain-type  domain-type
    ++-ro domain  domain
++-ro path-key {path-key}?
  ++-ro enabled?  boolean
  ++-ro discard-timer?  uint32
  ++-ro reuse-time?  uint32
  ++-ro pce-id?  inet:ip-address
++-ro (auth-type-selection)?
  +--:(auth-key-chain)
    ++-ro key-chain?  key-chain:key-chain-ref
  +--:(auth-key)
    ++-ro key?  string
++-ro crypto-algorithm
  +--:(algorithm)?
    +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
      ++-ro hmac-sha-12?  empty
    +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
      ++-ro aes-cmac-prf-128?  empty
    +--:(md5)
      ++-ro md5?  empty
    +--:(sha-1)
      ++-ro sha-1?  empty
    +--:(hmac-sha-1)
      ++-ro hmac-sha-1?  empty
    +--:(hmac-sha-256)
      ++-ro hmac-sha-256?  empty
    +--:(hmac-sha-384)
      ++-ro hmac-sha-384?  empty
    +--:(hmac-sha-512)
      ++-ro hmac-sha-512?  empty
    +--:(clear-text) {clear-text}?
      ++-ro clear-text?  empty
    +--:(replay-protection-only) {replay-protection-only}?
      ++-ro replay-protection-only?  empty
  +--:(auth-tls) {tls}?
    ++-ro tls
    ++-ro connect-timer?  uint32
    ++-ro connect-max-retry?  uint32
    ++-ro init-backoff-timer?  uint32
    ++-ro max-backoff-timer?  uint32
    ++-ro open-wait-timer?  uint32
    ++-ro keep-wait-timer?  uint32
    ++-ro keep-alive-timer?  uint32
    ++-ro dead-timer?  uint32
    ++-ro allow-negotiation?  boolean
    ++-ro max-keep-alive-timer?  uint32
++--ro global-source  -> /pcep-state/entity/lsp-db/associatio
++--ro extended-id   -> /pcep-state/entity/lsp-db/associatio
++--ro path-keys {path-key}?
  ++--ro path-keys* [path-key]
    ++--ro path-key         uint16
    ++--ro cps
      ++--ro explicit-route-objects* [index]
        ++--ro index         uint8
        ++--ro explicit-route-usage? identityref
        ++--ro (type)?
          ---:(ipv4-address)
            ++--ro v4-address?    inet:ipv4-address
            ++--ro v4-prefix-length? uint8
            ++--ro v4-loose?        boolean
          ---:(ipv6-address)
            ++--ro v6-address?    inet:ipv6-address
            ++--ro v6-prefix-length? uint8
            ++--ro v6-loose?        boolean
          ---:(as-number)
            ++--ro as-number?      uint16
          ---:(unnumbered-link)
            ++--ro router-id?      inet:ip-address
            ++--ro interface-id?   uint32
          ---:(label)
            ++--ro value?          uint32
        ++--ro pcc-original?  -> /pcep-state/entity/peers/peer/addr
        ++--ro req-id?         uint32
        ++--ro retrieved?      boolean
        ++--ro pcc-retrieved?  -> /pcep-state/entity/peers/peer/addr
        ++--ro creation-time?  yang:timestamp
        ++--ro discard-time?   uint32
        ++--ro reuse-time?     uint32
        ++--ro of-list {objective-function}?
          ++--ro objective-function* [of]
            ++--ro of   objective-function
        ++--ro peers
          ++--ro peer* [addr]
            ++--ro addr       inet:ip-address
            ++--ro role?      pcep-role
          ++--ro domain
            ++--ro domain* [domain-type domain]
              ++--ro domain-type domain-type
            ++--ro domain    domain
          ++--ro capability
            ++--ro gmpls?        boolean {gmpls}? 
            ++--ro bi-dir?       boolean
            ++--ro diverse?      boolean
            ++--ro load-balance? boolean
++--ro synchronize?    boolean {svec}? 
++--ro objective-function?    boolean {objective-function}? 
++--ro add-path-constraint?    boolean 
++--ro prioritization?    boolean 
++--ro multi-request?    boolean 
++--ro gco?    boolean {gco}? 
++--ro p2mp?    boolean {p2mp}? 
++--ro stateful {stateful}? 
|    ++--ro enabled?    boolean 
++--ro active?    boolean 
++--ro pce-initiated?    boolean {pce-initiated}? 
++--ro include-db-ver?    boolean {stateful-sync-opt}? 
++--ro trigger-resync?    boolean {stateful-sync-opt}? 
++--ro trigger-initial-sync?    boolean {stateful-sync-opt}? 
++--ro incremental-sync?    boolean {stateful-sync-opt}? 
++--ro sr {sr}? 
    ++--ro enabled?    boolean 
++--ro pce-info 
    ++--ro scope 
|    ++--ro intra-area-scope?    boolean 
++--ro intra-area-pref?    uint8 
++--ro inter-area-scope?    boolean 
++--ro inter-area-scope-default?    boolean 
++--ro inter-area-pref?    uint8 
++--ro inter-as-scope?    boolean 
++--ro inter-as-scope-default?    boolean 
++--ro inter-as-pref?    uint8 
++--ro inter-layer-scope?    boolean 
++--ro inter-layer-pref?    uint8 
++--ro neigh-domains 
    ++--ro domain* [domain-type domain] 
|    ++--ro domain-type    domain-type 
    ++--ro domain    domain 
++--ro delegation-pref?    uint8 {stateful}? 
++--ro (auth-type-selection)? 
++--:(auth-key-chain) 
|    ++--ro key-chain?    key-chain:key-chain-ref 
++--:(auth-key) 
    ++--ro key?    string 
++--ro crypto-algorithm 
    ++--ro (algorithm)? 
        ++--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}? 
|    ++--ro hmac-shal-12?    empty 
        ++--:(aes-cmac-prf-128) {aes-cmac-prf-128}? 
|    ++--ro aes-cmac-prf-128?    empty 
        ++--:(md5) 
|    ++--ro md5?    empty 
        ++--:(sha-1)
---ro sha-1?               empty
  +--:(hmac-sha-1)               empty
  | +--ro hmac-sha-1?             empty
  +--:(hmac-sha-256)              empty
  | +--ro hmac-sha-256?           empty
  +--:(hmac-sha-384)              empty
  | +--ro hmac-sha-384?           empty
  +--:(hmac-sha-512)              empty
  | +--ro hmac-sha-512?           empty
  +--:(clear-text) {clear-text}?               empty
  +--:(replay-protection-only) (replay-protection-only)

---ro replay-protection-only?               empty
  +--:(auth-tls) {tls}?               empty
  | +--ro tls
  +--ro of-list {objective-function}?
    +--ro objective-function* [of]
      +--ro of objective-function
        +--ro discontinuity-time? yang:timestamp
        +--ro initiate-session? boolean
        +--ro session-exists? boolean
        +--ro num-sess-setup-ok? yang:counter32
        +--ro num-sess-setup-fail? yang:counter32
        +--ro session-up-time? yang:timestamp
        +--ro session-fail-time? yang:timestamp
        +--ro session-fail-up-time? yang:timestamp
    +--ro pcep-stats
      +--ro avg-rsp-time? uint32
      +--ro lwm-rsp-time? uint32
      +--ro hwm-rsp-time? uint32
      +--ro num-pcreq-sent? yang:counter32
      +--ro num-pcreq-rcvd? yang:counter32
      +--ro num-pcrep-sent? yang:counter32
      +--ro num-pcrep-rcvd? yang:counter32
      +--ro num-pcerr-sent? yang:counter32
      +--ro num-pcerr-rcvd? yang:counter32
      +--ro num-pcntf-sent? yang:counter32
      +--ro num-pcntf-rcvd? yang:counter32
      +--ro num-keepalive-sent? yang:counter32
      +--ro num-keepalive-rcvd? yang:counter32
      +--ro num-unknown-rcvd? yang:counter32
      +--ro num-corrupt-rcvd? yang:counter32
      +--ro num-req-sent? yang:counter32
      +--ro num-req-sent-pend-rep? yang:counter32
      +--ro num-req-sent-ero-rcvd? yang:counter32
      +--ro num-req-sent-nopath-rcvd? yang:counter32
      +--ro num-req-sent-cancel-rcvd? yang:counter32
      +--ro num-req-sent-error-rcvd? yang:counter32
++ ro num-req-sent-timeout? yang:counter32
++ ro num-req-sent-cancel-sent? yang:counter32
++ ro num-req-rcvd? yang:counter32
++ ro num-req-rcvd-pend-rep? yang:counter32
++ ro num-req-rcvd-ero-sent? yang:counter32
++ ro num-req-rcvd-nopath-sent? yang:counter32
++ ro num-req-rcvd-cancel-sent? yang:counter32
++ ro num-req-rcvd-error-sent? yang:counter32
++ ro num-req-rcvd-cancel-rcvd? yang:counter32
++ ro num-req-rcvd-unknown? yang:counter32
++ ro num-req-rcvd-unknown? yang:counter32
++ ro svec {svec}?
  ++ ro num-svec-sent? yang:counter32
  ++ ro num-svec-req-sent? yang:counter32
  ++ ro num-svec-rcvd? yang:counter32
  ++ ro num-svec-req-rcvd? yang:counter32
++ ro stateful {stateful}?
  ++ ro num-pcrpt-sent? yang:counter32
  ++ ro num-pcrpt-rcvd? yang:counter32
  ++ ro num-pcupd-sent? yang:counter32
  ++ ro num-pcupd-rcvd? yang:counter32
  ++ ro num-rpt-sent? yang:counter32
  ++ ro num-rpt-rcvd? yang:counter32
  ++ ro num-rpt-rcvd-error-sent? yang:counter32
  ++ ro num-upd-sent? yang:counter32
  ++ ro num-upd-rcvd? yang:counter32
  ++ ro num-upd-rcvd-error-sent? yang:counter32
  ++ ro num-upd-rcvd-unknown? yang:counter32
  ++ ro num-upd-rcvd-undelegated? yang:counter32
  ++ ro num-upd-rcvd-error-sent? yang:counter32
++ ro initiation {pce-initiated}?
  ++ ro num-pcinitiate-sent? yang:counter32
  ++ ro num-pcinitiate-rcvd? yang:counter32
  ++ ro num-initiate-sent? yang:counter32
  ++ ro num-initiate-rcvd? yang:counter32
  ++ ro num-initiate-rcvd-error-sent? yang:counter32
++ ro path-key {path-key}?
  ++ ro num-unknown-path-key? yang:counter32
  ++ ro num-exp-path-key? yang:counter32
  ++ ro num-dup-path-key? yang:counter32
  ++ ro num-path-key-no-attempt? yang:counter32
++ ro num-req-sent-closed? yang:counter32
++ ro num-req-rcvd-closed? yang:counter32
++ ro sessions
  ++ ro session* [initiator]
    ++ ro initiator pcep-initiator
    ++ ro state-last-change? pcep:timestamp
    ++ ro state? pcep:sess-state
    ++ ro session-creation? pcep:timestamp
++--ro num-req-rcvd-error-sent?  yang:counter32
++--ro num-req-rcvd-cancel-rcvd?  yang:counter32
++--ro num-rep-rcvd-unknown?  yang:counter32
++--ro num-req-rcvd-unknown?  yang:counter32
++--ro svec {svec}?
  | ++--ro num-svec-sent?  yang:counter32
  | ++--ro num-svec-req-sent?  yang:counter32
  | ++--ro num-svec-rcvd?  yang:counter32
  | ++--ro num-svec-req-rcvd?  yang:counter32
++--ro stateful {stateful}?
  | ++--ro num-pcrpt-sent?  yang:counter32
  | ++--ro num-pcrpt-rcvd?  yang:counter32
  | ++--ro num-pcupd-sent?  yang:counter32
  | ++--ro num-pcupd-rcvd?  yang:counter32
  | ++--ro num-rpt-rcvd-error-sent?  yang:counter32
  | ++--ro num-upd-sent?  yang:counter32
  | ++--ro num-upd-rcvd?  yang:counter32
  | ++--ro num-upd-rcvd-unknown?  yang:counter32
  | ++--ro num-upd-rcvd-undelegated?  yang:counter32
  | ++--ro num-upd-rcvd-error-sent?  yang:counter32
++--ro initiation {pce-initiated}?
  | ++--ro num-pcinitiate-sent?  yang:counter32
  | ++--ro num-pcinitiate-rcvd?  yang:counter32
  | ++--ro num-initiate-sent?  yang:counter32
  | ++--ro num-initiate-rcvd?  yang:counter32
  | ++--ro num-initiate-rcvd-error-sent?  yang:counter32
++--ro path-key {path-key}?
  | ++--ro num-unknown-path-key?  yang:counter32
  | ++--ro num-exp-path-key?  yang:counter32
  | ++--ro num-dup-path-key?  yang:counter32
  | ++--ro num-path-key-no-attempt?  yang:counter32

notifications:
  +---n pcep-session-up
    | ++--ro peer-addr?  -> /pcep-state/entity/peers/peer/addr
    | ++--ro session-initiator?  -> /pcep-state/entity/peers/peer/sessions/session/initiator
    | ++--ro state-last-change?  yang:timestamp
    | ++--ro state?  pcep-sess-state
  +---n pcep-session-down
    | ++--ro peer-addr?  -> /pcep-state/entity/peers/peer/addr
    | ++--ro session-initiator?  pcep-initiator
    | ++--ro state-last-change?  yang:timestamp
    | ++--ro state?  pcep-sess-state
  +---n pcep-session-local-overload
    | ++--ro peer-addr?  -> /pcep-state/entity/peers/peer/addr
    | ++--ro session-initiator?  -> /pcep-state/entity/peers/peer/sessions/session/initiator
    | ++--ro overloaded?  boolean

5.1. The Entity

The PCEP yang module may contain status information for the local PCEP entity.

The entity has an IP address (using ietf-inet-types [RFC6991]) and a "role" leaf (the local entity PCEP role) as mandatory.

Note that, the PCEP MIB module [RFC7420] uses an entity list and a system generated entity index as a primary index to the read only entity table. If the device implements the PCEP MIB, the "index" leaf MUST contain the value of the corresponding pcePcepEntityIndex and only one entity is assumed.

5.2. The Peer Lists

The peer list contains peer(s) that the local PCEP entity knows about. A PCEP speaker is identified by its IP address. If there is a PCEP speaker in the network that uses multiple IP addresses then it looks like multiple distinct peers to the other PCEP speakers in the network.

Since PCEP sessions can be ephemeral, the peer list tracks a peer even when no PCEP session currently exists to that peer. The statistics contained are an aggregate of the statistics for all successive sessions to that peer.

To limit the quantity of information that is stored, an implementation MAY choose to discard this information if and only if no PCEP session exists to the corresponding peer.

The data model for PCEP peer presented in this document uses a flat list of peers. Each peer in the list is identified by its IP address (addr-type, addr).
There is one list for static peer configuration ("/pcep/entity/peers"), and a separate list for the operational state of all peers (i.e. static as well as discovered)("/pcep-state/entity/peers"). The former is used to enable remote PCE configuration at PCC (or PCE) while the latter has the operational state of these peers as well as the remote PCE peer which were discovered and PCC peers that have initiated session.

5.3. The Session Lists

The session list contains PCEP session that the PCEP entity (PCE or PCC) is currently participating in. The statistics in session are semantically different from those in peer since the former applies to the current session only, whereas the latter is the aggregate for all sessions that have existed to that peer.

Although [RFC5440] forbids more than one active PCEP session between a given pair of PCEP entities at any given time, there is a window during session establishment where two sessions may exist for a given pair, one representing a session initiated by the local PCEP entity and the other representing a session initiated by the peer. If either of these sessions reaches active state first, then the other is discarded.

The data model for PCEP session presented in this document uses a flat list of sessions. Each session in the list is identified by its initiator. This index allows two sessions to exist transiently for a given peer, as discussed above.

There is only one list for the operational state of all sessions ("/pcep-state/entity/peers/peer/sessions/session").

5.4. Notifications

This YANG model defines a list of notifications to inform client of important events detected during the protocol operation. The notifications defined cover the PCEP MIB notifications.

6. Advanced PCE Features

This document contains a specification of the base PCEP YANG module, "ietf-pcep" which provides the basic PCEP [RFC5440] data model.

This document further handles advanced PCE features like -
- Capability and Scope
- Domain information (local/neighbour)
6.1. Stateful PCE’s LSP-DB

In the operational state of PCEP which supports stateful PCE mode, the list of LSP state are maintained in LSP-DB. The key is the PLSP-ID and the PCC IP address.

The PCEP data model contains the operational state of LSPs (pcep-state/entity/lsp-db/lsp/) with PCEP specific attributes. The generic TE attributes of the LSP are defined in [I-D.ietf-teas-yang-te]. A reference to LSP state in TE model is maintained.

7. Open Issues and Next Step

This section is added so that open issues can be tracked. This section would be removed when the document is ready for publication.

7.1. The PCE-Initiated LSP

The TE Model at [I-D.ietf-teas-yang-te] should support creating tunnels at the controller (PCE) and marking them as PCE-Initiated. The LSP-DB in the PCEP Yang (pcep-state/entity/lsp-db/lsp/initiation) also marks the LSPs which are PCE-initiated.

7.2. PCEP over TLS (PCEPS)

A future version of this document would add TLS related configurations.
8. PCEP YANG Module

RFC Ed.: In this section, replace all occurrences of ‘XXXX’ with the actual RFC number and all occurrences of the revision date below with the date of RFC publication (and remove this note).

<CODE BEGINS> file "ietf-pcep@2016-10-27.yang"
module ietf-pcep {  
namespace "urn:ietf:params:xml:ns:yang:ietf-pcep";  
prefix pcep;  

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

import ietf-yang-types {  
    prefix "yang";  
}  

import ietf-te {  
    prefix "te";  
}  

import ietf-te-types {  
    prefix "te-types";  
}  

import ietf-key-chain {  
    prefix "key-chain";  
}  

organization  
    "IETF PCE (Path Computation Element) Working Group";  

contact  
    "WG Web: <http://tools.ietf.org/wg/pce/>  
    WG List: <mailto:pce@ietf.org>  
    WG Chair: JP Vasseur  
        <mailto:jpv@cisco.com>  
    WG Chair: Julien Meuric  
        <mailto:julien.meuric@orange.com>  
    WG Chair: Jonathan Hardwick  
        <mailto:Jonathan.Hardwick@metaswitch.com>  
    Editor: Dhruv Dhody  
        <mailto:dhruv.ietf@gmail.com>";  

description
"The YANG module defines a generic configuration and
operational model for PCEP common across all of the
vendor implementations."

revision 2016-10-27 {
  description "Initial revision.";
  reference
    "RFC XXXX:  A YANG Data Model for Path Computation
    Element Communications Protocol (PCEP)"
}

/*
 * Identities
 */

identity pcep {
  description "Identity for the PCEP protocol.";
}

/*
 * Typedefs
 */
typedef pcep-role {
  type enumeration {
    enum unknown {
      value "0";
      description
        "An unknown role";
    }
    enum pcc {
      value "1";
      description
        "The role of a Path Computation Client";
    }
    enum pce {
      value "2";
      description
        "The role of Path Computation Element";
    }
    enum pcc-and-pce {
      value "3";
      description
        "The role of both Path Computation Client and
        Path Computation Element";
    }
  }
}
description
"The role of a PCEP speaker. Takes one of the following values
- unknown(0): the role is not known.
- pcc(1): the role is of a Path Computation Client (PCC).
- pce(2): the role is of a Path Computation Server (PCE).
- pccAndPce(3): the role is of both a PCC and a PCE."

typedef pcep-admin-status {
  type enumeration {
    enum admin-status-up {
      value "1";
      description
      "Admin Status is Up";
    }
    enum admin-status-down {
      value "2";
      description
      "Admin Status is Down";
    }
  }
}

description
"The Admin Status of the PCEP entity. Takes one of the following values
- admin-status-up(1): Admin Status is Up.
- admin-status-down(2): Admin Status is Down"

typedef pcep-oper-status {
  type enumeration {
    enum oper-status-up {
      value "1";
      description
      "The PCEP entity is active";
    }
    enum oper-status-down {
      value "2";
      description
      "The PCEP entity is inactive";
    }
  }
}
enum oper-status-going-up {
  value "3";
  description
    "The PCEP entity is activating";
}
enum oper-status-going-down {
  value "4";
  description
    "The PCEP entity is deactivating";
}
enum oper-status-failed {
  value "5";
  description
    "The PCEP entity has failed and will recover
    when possible.";
}
enum oper-status-failed-perm {
  value "6";
  description
    "The PCEP entity has failed and will not recover
    without operator intervention";
}

description
  "The operational status of the PCEP entity.
  Takes one of the following values
  - oper-status-up(1): Active
  - oper-status-down(2): Inactive
  - oper-status-going-up(3): Activating
  - oper-status-going-down(4): Deactivating
  - oper-status-failed(5): Failed
  - oper-status-failed-perm(6): Failed Permanently";
}
typedef pcep-initiator {
  type enumeration {
    enum local {
      value "1";
      description
        "The local PCEP entity initiated the session";
    }
    enum remote {
      value "2";
      description
        "The remote PCEP peer initiated the session";
    }
  }
}
typedef pcep-sess-state {
    type enumeration {
        enum tcp-pending {
            value "1";
            description "The tcp-pending state of PCEP session.";
        }
        enum open-wait {
            value "2";
            description "The open-wait state of PCEP session.";
        }
        enum keep-wait {
            value "3";
            description "The keep-wait state of PCEP session.";
        }
        enum session-up {
            value "4";
            description "The session-up state of PCEP session.";
        }
    }
}

description "The current state of the session.
The set of possible states excludes the idle state
since entries do not exist in the idle state.
Takes one of the following values
    - tcp-pending(1): PCEP TCP Pending state
    - open-wait(2): PCEP Open Wait state
    - keep-wait(3): PCEP Keep Wait state
    - session-up(4): PCEP Session Up state";

typedef domain-type {
    type enumeration {

enum ospf-area {
  value "1";
  description
    "The OSPF area."
};
enum isis-area {
  value "2";
  description
    "The IS-IS area."
};
enum as {
  value "3";
  description
    "The Autonomous System (AS)."
};
description
  "The PCE Domain Type";

typedef domain-ospf-area {
  type union {
    type uint32;
    type yang:dotted-quad;
  }
  description
    "OSPF Area ID."
};
typedef domain-isis-area {
  type string {
    pattern '[0-9A-Fa-f]{2}\.(\[0-9A-Fa-f]{4}\.){0,3}';
  }
  description
    "IS-IS Area ID."
};
typedef domain-as {
  type uint32;
  description
    "Autonomous System number."
};
typedef domain {
  type union {
    type domain-ospf-area;
    type domain-isis-area;
  }
}
type domain-as;
}
description
"The Domain Information";
}
typedef operational-state {
  type enumeration {
    enum down {
      value "0";
      description
      "not active.";
    }
    enum up {
      value "1";
      description
      "signalled.";
    }
    enum active {
      value "2";
      description
      "up and carrying traffic.";
    }
    enum going-down {
      value "3";
      description
      "LSP is being torn down, resources are
      being released.";
    }
    enum going-up {
      value "4";
      description
      "LSP is being signalled.";
    }
  }
  description
  "The operational status of the LSP";
}
typedef lsp-error {
  type enumeration {
    enum no-error {
      value "0";
      description
      "No error, LSP is fine.";
    }
    enum unknown {
      value "1";
    }
  }
  description
  "LSP error code";
}
enum limit {
  value "2",
  description "Limit reached for PCE-controlled LSPs."
}
enum pending {
  value "3",
  description "Too many pending LSP update requests."
}
enum unacceptable {
  value "4",
  description "Unacceptable parameters."
}
enum internal {
  value "5",
  description "Internal error."
}
enum admin {
  value "6",
  description "LSP administratively brought down."
}
enum preempted {
  value "7",
  description "LSP preempted."
}
enum rsvp {
  value "8",
  description "RSVP signaling error."
}

typedef sync-state {
  type enumeration {
    enum pending {
      value "0",
      description

"The state synchronization has not started."
}
enum ongoing {
  value "1";
  description
    "The state synchronization is ongoing.";
}
enum finished {
  value "2";
  description
    "The state synchronization is finished.";
}

description
  "The LSP-DB state synchronization operational status."
}
typedef pst{
  type enumeration{
    enum rsvp-te{
      value "0";
      description
        "RSVP-TE signaling protocol";
    }
    enum sr{
      value "1";
      description
        "Segment Routing Traffic Engineering";
    }
  }
  description
    "The Path Setup Type";
}
typedef assoc-type{
  type enumeration{
    enum protection{
      value "1";
      description
        "Path Protection Association Type";
    }
  }
  description
    "The PCEP Association Type";
}
typedef objective-function{
    type enumeration{
        enum mcp{
            value "1";
            description
                "Minimum Cost Path (MCP)";
        }
        enum mlp{
            value "2";
            description
                "Minimum Load Path (MLP)";
        }
        enum mbp{
            value "3";
            description
                "Maximum residual Bandwidth Path (MBP)";
        }
        enum mbc{
            value "4";
            description
                "Minimize aggregate Bandwidth Consumption (MBC)";
        }
        enum mll{
            value "5";
            description
                "Minimize the Load of the most loaded Link (MLL)";
        }
        enum mcc{
            value "6";
            description
                "Minimize the Cumulative Cost of a set of paths (MCC)";
        }
        enum spt{
            value "7";
            description
                "Shortest Path Tree (SPT)";
        }
        enum mct{
            value "8";
            description
                "Minimum Cost Tree (MCT)";
        }
        enum mplp{
            value "9";
            description
                "Minimum Packet Loss Path (MPLP)";
        }
    }
}
enum mup {
    value "10";
    description "Maximum Under-Utilized Path (MUP)";
}
enum mrup {
    value "11";
    description "Maximum Reserved Under-Utilized Path (MRUP)";
}
description "The PCEP Objective functions";

/*
 * Features
 */
feature svec {
    description "Support synchronized path computation.";
}
feature gmpls {
    description "Support GMPLS.";
}
feature objective-function {
    description "Support OF as per RFC 5541.";
}
feature gco {
    description "Support GCO as per RFC 5557.";
}
feature path-key {
    description "Support path-key as per RFC 5520.";
}
feature p2mp {
    description "Support P2MP as per RFC 6006.";
}

feature stateful {
    description
        "Support stateful PCE.";
}

feature stateful-sync-opt {
    description
        "Support stateful sync optimization";
}

feature pce-initiated {
    description
        "Support PCE-Initiated LSP.";
}

feature tls {
    description
        "Support PCEP over TLS.";
}

feature sr {
    description
        "Support Segment Routing for PCE.";
}

/*
 * Groupings
 */

grouping pcep-entity-info{
    description
        "This grouping defines the attributes for PCEP entity.";
    leaf connect-timer {
        type uint32 {
            range "1..65535";
        }
    }
    units "seconds";
    default 60;
    description
        "The time in seconds that the PCEP entity will wait to establish a TCP connection with a peer. If a TCP connection is not established within this time then PCEP aborts the session setup attempt.";
    reference
        "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}
leaf connect-max-retry {
  type uint32;
  default 5;
  description
  "The maximum number of times the system tries to establish a TCP connection to a peer before the session with the peer transitions to the idle state."
  reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"
}

leaf init-backoff-timer {
  type uint32 { range "1..65535"; }
  units "seconds";
  description
  "The initial back-off time in seconds for retrying a failed session setup attempt to a peer. The back-off time increases for each failed session setup attempt, until a maximum back-off time is reached. The maximum back-off time is max-backoff-timer.";
}

leaf max-backoff-timer {
  type uint32;
  units "seconds";
  description
  "The maximum back-off time in seconds for retrying a failed session setup attempt to a peer. The back-off time increases for each failed session setup attempt, until this maximum value is reached. Session setup attempts then repeat periodically without any further increase in back-off time.";
}

leaf open-wait-timer {
  type uint32 { range "1..65535"; }
  units "seconds";
  default 60;
  description
  "The maximum number of times the system tries to establish a TCP connection to a peer before the session with the peer transitions to the idle state."
  reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
"The time in seconds that the PCEP entity will wait to receive an Open message from a peer after the TCP connection has come up. If no Open message is received within this time then PCEP terminates the TCP connection and deletes the associated sessions.";

reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";

leaf keep-wait-timer {
  type uint32 {
    range "1..65535";
  }
  units "seconds";
  default 60;
  description
  "The time in seconds that the PCEP entity will wait to receive a Keepalive or PCErr message from a peer during session initialization after receiving an Open message. If no Keepalive or PCErr message is received within this time then PCEP terminates the TCP connection and deletes the associated sessions.";

  reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";
}

leaf keep-alive-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  default 30;
  description
  "The keep alive transmission timer that this PCEP entity will propose in the initial OPEN message of each session it is involved in. This is the maximum time between two consecutive messages sent to a peer. Zero means that the PCEP entity prefers not to send Keepalives at all. Note that the actual Keepalive transmission intervals, in either direction of an active PCEP session, are determined by negotiation between the peers as specified by RFC 5440, and so may differ from this configured value.";

leaf dead-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  must "{. > ../keep-alive-timer}" {
    error-message "The dead timer must be " + "larger than the keep alive timer";
    description "This value MUST be greater than keep-alive-timer.";
  }
  default 120;
  description "The dead timer that this PCEP entity will propose in the initial OPEN message of each session it is involved in. This is the time after which a peer should declare a session down if it does not receive any PCEP messages. Zero suggests that the peer does not run a dead timer at all.";
  reference "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}

leaf allow-negotiation{
  type boolean;
  description "Whether the PCEP entity will permit negotiation of session parameters.";
}

leaf max-keep-alive-timer{
  type uint32 {
    range "0..255";
  }
  units "seconds";
  description "In PCEP session parameter negotiation in seconds, the maximum value that this PCEP entity will accept from a peer for the interval between Keepalive transmissions. Zero means that the PCEP
entity will allow no Keepalive transmission at all."
}

leaf max-dead-timer{
  type uint32 {
    range "0..255";
  }
  units "seconds";
  description
  "In PCEP session parameter negotiation in seconds, the maximum value that this PCEP entity will accept from a peer for the Dead timer. Zero means that the PCEP entity will allow not running a Dead timer.";
}

leaf min-keep-alive-timer{
  type uint32 {
    range "0..255";
  }
  units "seconds";
  description
  "In PCEP session parameter negotiation in seconds, the minimum value that this PCEP entity will accept for the interval between Keepalive transmissions. Zero means that the PCEP entity insists on no Keepalive transmission at all.";
}

leaf min-dead-timer{
  type uint32 {
    range "0..255";
  }
  units "seconds";
  description
  "In PCEP session parameter negotiation in seconds, the minimum value that this PCEP entity will accept for the Dead timer. Zero means that the PCEP entity insists on not running a Dead timer.";
}

leaf sync-timer{
  if-feature svec;
  type uint32 {
    range "0..65535";
  }
}

units "seconds";
default 60;
description
  "The value of SyncTimer in seconds is used in the case of synchronized path computation request using the SVEC object. Consider the case where a PCReq message is received by a PCE that contains the SVEC object referring to M synchronized path computation requests. If after the expiration of the SyncTimer all the M path computation requests have not been, received a protocol error is triggered and the PCE MUST cancel the whole set of path computation requests. The aim of the SyncTimer is to avoid the storage of unused synchronized requests should one of them get lost for some reasons (for example, a misbehaving PCC). Zero means that the PCEP entity does not use the SyncTimer."
reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"
}

leaf request-timer{
  type uint32 {
    range "1..65535";
  }
  units "seconds";
  description
    "The maximum time that the PCEP entity will wait for a response to a PCReq message."
}

leaf max-sessions{
  type uint32;
  description
    "Maximum number of sessions involving this PCEP entity that can exist at any time."
}

leaf max-unknown-reqs{
  type uint32;
  default 5;
  description
    "The maximum number of unrecognized requests and replies that any session on this PCEP entity is
willing to accept per minute before terminating the session.
A PCRep message contains an unrecognized reply if it contains an RP object whose request ID does not correspond to any in-progress request sent by this PCEP entity.
A PCReq message contains an unrecognized request if it contains an RP object whose request ID is zero."

reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
}

leaf max-unknown-msgs{
  type uint32;
  default 5;
  description
  "The maximum number of unknown messages that any session on this PCEP entity is willing to accept per minute before terminating the session.";
  reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
}

}//pcep-entity-info
grouping pce-scope{
  description
  "This grouping defines PCE path computation scope information which maybe relevant to PCE selection. This information corresponds to PCE auto-discovery information."
  reference
  "RFC 5088: OSPF Protocol Extensions for Path Computation Element (PCE) Discovery"
  "RFC 5089: IS-IS Protocol Extensions for Path Computation Element (PCE) Discovery"

  leaf intra-area-scope{
    type boolean;
    default true;
    description
    "PCE can compute intra-area paths."
  }

  leaf intra-area-pref{
type uint8{
    range "0..7";
}

description
    "The PCE’s preference for intra-area TE LSP computation.";

leaf inter-area-scope{
    type boolean;
    default false;
    description
        "PCE can compute inter-area paths.";
}

leaf inter-area-scope-default{
    type boolean;
    default false;
    description
        "PCE can act as a default PCE for inter-area path computation.";
}

leaf inter-area-pref{
    type uint8{
        range "0..7";
    }
    description
        "The PCE’s preference for inter-area TE LSP computation.";
}

leaf inter-as-scope{
    type boolean;
    default false;
    description
        "PCE can compute inter-AS paths.";
}

leaf inter-as-scope-default{
    type boolean;
    default false;
    description
        "PCE can act as a default PCE for inter-AS path computation.";
}

leaf inter-as-pref{
    type uint8{
        range "0..7";
    }
    description
        "The PCE’s preference for inter-AS TE LSP computation.";
leaf inter-layer-scope{
  type boolean;
  default false;
  description
       "PCE can compute inter-layer paths.";
}
leaf inter-layer-pref{
  type uint8{
    range "0..7";
  }
  description
       "The PCE’s preference for inter-layer TE LSP
        computation.";
}
//pce-scope

//domain

grouping domain{
  description
       "This grouping specifies a Domain where the
        PCEP speaker has topology visibility.";
  leaf domain-type{
    type domain-type;
    description
       "The domain type.";
  }
  leaf domain{
    type domain;
    description
       "The domain Information.";
  }
}//domain

//capability

grouping capability{
  description
       "This grouping specifies a capability
        information of local PCEP entity. This may
        be relevant to PCE selection as well. This
        information corresponds to PCE auto-discovery
        information.";
  reference
       "RFC 5088: OSPF Protocol Extensions for Path
        Computation Element (PCE) Discovery
        RFC 5089: IS-IS Protocol Extensions for Path
        Computation Element (PCE) Discovery";
  leaf gmpls{
if-feature gmpls;
  type boolean;
  description
    "Path computation with GMPLS link constraints.";
}
leaf bi-dir{
  type boolean;
  description
    "Bidirectional path computation.";
}
leaf diverse{
  type boolean;
  description
    "Diverse path computation.";
}
leaf load-balance{
  type boolean;
  description
    "Load-balanced path computation.";
}
leaf synchronize{
  if-feature svec;
  type boolean;
  description
    "Synchronized paths computation.";
}
leaf objective-function{
  if-feature objective-function;
  type boolean;
  description
    "Support for multiple objective functions.";
}
leaf add-path-constraint{
  type boolean;
  description
    "Support for additive path constraints (max hop count, etc.).";
}
leaf prioritization{
  type boolean;
  description
    "Support for request prioritization.";
}
leaf multi-request{
  type boolean;
  description
    "Support for multiple requests per message.";
leaf gco{
    if-feature gco;
    type boolean;
    description
        "Support for Global Concurrent Optimization (GCO).";
}
leaf p2mp{
    if-feature p2mp;
    type boolean;
    description
        "Support for P2MP path computation.";
}

container stateful{
    if-feature stateful;
    description
        "If stateful PCE feature is present";
    leaf enabled{
        type boolean;
        description
            "Enabled or Disabled";
    }
    leaf active{
        type boolean;
        description
            "Support for active stateful PCE.";
    }
    leaf pce-initiated{
        if-feature pce-initiated;
        type boolean;
        description
            "Support for PCE-initiated LSP.";
    }
    leaf include-db-ver{
        if-feature stateful-sync-opt;
        type boolean;
        description
            "Support inclusion of LSP-DB-VERSION in LSP object";
    }
    leaf trigger-resync{
        if-feature stateful-sync-opt;
        type boolean;
        description
            "Support PCE triggered re-synchronization";
    }
}
leaf trigger-initial-sync{
    if-feature stateful-sync-opt;
    type boolean;
    description
    "PCE triggered initial synchronization";
}
leaf incremental-sync{
    if-feature stateful-sync-opt;
    type boolean;
    description
    "Support incremental (delta) sync";
}
}
container sr{
    if-feature sr;
    description
    "If segment routing is supported";
    leaf enabled{
        type boolean;
        description
        "Enabled or Disabled";
    }
}
}
}//capability
grouping info{
    description
    "This grouping specifies all information which maybe relevant to both PCC and PCE. 
    This information corresponds to PCE auto-discovery information.";
    container domain{
        description
        "The local domain for the PCEP entity";
        list domain{
            key "domain-type domain";
            description
            "The local domain.";
            uses domain{
                description
                "The local domain for the PCEP entity.";
            }
        }
    }
    container capability{
        description
        "The PCEP entity capability";
    }

uses capability{
    description
        "The PCEP entity supported capabilities.";
}

//info

grouping pce-info{
    description
        "This grouping specifies all PCE information which maybe relevant to the PCE selection. This information corresponds to PCE auto-discovery information.";
    container scope{
        description
            "The path computation scope";
        uses pce-scope;
    }
    container neigh-domains{
        description
            "The list of neighbour PCE-Domain toward which a PCE can compute paths";
        list domain{
            key "domain-type domain";
            description
                "The neighbour domain.";
            uses domain{
                description
                    "The PCE neighbour domain.";
            }
        }
    }
}

//pce-info

grouping pcep-stats{
    description
        "This grouping defines statistics for PCEP. It is used for both peer and current session.";
    leaf avg-rsp-time{
        type uint32;
        units "milliseconds";
        must "(/pcep-state/entity/peers/peer/role != 'pcc' +
" or " +
"{(/pcep-state/entity/peers/peer/role = ‘pcc’" +
" and (. = 0))})" {
  error-message
  "Invalid average response time";
  description
  "If role is pcc then this leaf is meaningless
   and is set to zero.";
}
description
"The average response time.
If an average response time has not been
calculated then this leaf has the value zero.";
}

leaf lwm-rsp-time{
  type uint32;
  units "milliseconds";
  must "{(/pcep-state/entity/peers/peer/role != ‘pcc’" +
  " or " +
  "(/pcep-state/entity/peers/peer/role = ‘pcc’" +
  " and (. = 0))})" {
    error-message
    "Invalid smallest (low-water mark)
     response time";
    description
    "If role is pcc then this leaf is meaningless
     and is set to zero.";
  }
description
"The smallest (low-water mark) response time seen.
If no responses have been received then this
leaf has the value zero.";
}

leaf hwm-rsp-time{
  type uint32;
  units "milliseconds";
  must "{(/pcep-state/entity/peers/peer/role != ‘pcc’" +
  " or " +
  "(/pcep-state/entity/peers/peer/role = ‘pcc’" +
  " and (. = 0))})" {
    error-message
    "Invalid greatest (high-water mark)
     response time seen";
    description
    "If role is pcc then this field is
     meaningless and is set to zero.";
}
leaf num-pcreq-sent{
  type yang:counter32;
  description
    "The number of PCReq messages sent.";
}

leaf num-pcreq-rcvd{
  type yang:counter32;
  description
    "The number of PCReq messages received.";
}

leaf num-pcrep-sent{
  type yang:counter32;
  description
    "The number of PCRep messages sent.";
}

leaf num-pcrep-rcvd{
  type yang:counter32;
  description
    "The number of PCRep messages received.";
}

leaf num-pcerr-sent{
  type yang:counter32;
  description
    "The number of PCErr messages sent.";
}

leaf num-pcerr-rcvd{
  type yang:counter32;
  description
    "The number of PCErr messages received.";
}

leaf num-pcntf-sent{
  type yang:counter32;
  description
    "The number of PCNTf messages sent.";
}
leaf num-pcntf-rcvd{
    type yang:counter32;
    description
    "The number of PCNtf messages received.";
}

leaf num-keepalive-sent{
    type yang:counter32;
    description
    "The number of Keepalive messages sent.";
}

leaf num-keepalive-rcvd{
    type yang:counter32;
    description
    "The number of Keepalive messages received.";
}

leaf num-unknown-rcvd{
    type yang:counter32;
    description
    "The number of unknown messages received.";
}

leaf num-corrupt-rcvd{
    type yang:counter32;
    description
    "The number of corrupted PCEP message received.";
}

leaf num-req-sent{
    type yang:counter32;
    description
    "The number of requests sent. A request corresponds
    1:1 with an RP object in a PCReq message. This might
    be greater than num-pcreq-sent because multiple
    requests can be batched into a single PCReq
    message.";
}

leaf num-req-sent-pend-rep{
    type yang:counter32;
    description
    "The number of requests that have been sent for
    which a response is still pending.";
}

leaf num-req-sent-ero-rcvd{
type yang:counter32;
description
"The number of requests that have been sent for
which a response with an ERO object was received.
Such responses indicate that a path was
successfully computed by the peer."
}

leaf num-req-sent-nopath-rcvd{
  type yang:counter32;
description
"The number of requests that have been sent for
which a response with a NO-PATH object was
received. Such responses indicate that the peer
could not find a path to satisfy the request."
}

leaf num-req-sent-cancel-rcvd{
  type yang:counter32;
description
"The number of requests that were cancelled with
a PCNtf message.
This might be different than num-pcntf-rcvd because
not all PCNtf messages are used to cancel requests,
and a single PCNtf message can cancel multiple
requests."
}

leaf num-req-sent-error-rcvd{
  type yang:counter32;
description
"The number of requests that were rejected with a
PCErr message.
This might be different than num-pcerr-rcvd because
not all PCErr messages are used to reject requests,
and a single PCErr message can reject multiple
requests."
}

leaf num-req-sent-timeout{
  type yang:counter32;
description
"The number of requests that have been sent to a peer
and have been abandoned because the peer has taken too
time to respond to them."
}
leaf num-req-sent-cancel-sent{
    type yang:counter32;
    description
        "The number of requests that were sent to the peer and
        explicitly cancelled by the local PCEP entity sending
        a PCNtf.";
}

leaf num-req-rcvd{
    type yang:counter32;
    description
        "The number of requests received. A request
        corresponds 1:1 with an RP object in a PCReq
        message. This might be greater than num-pcreq-rcvd because
        multiple requests can be batched into a single
        PCReq message.";
}

leaf num-req-rcvd-pend-rep{
    type yang:counter32;
    description
        "The number of requests that have been received for
        which a response is still pending.";
}

leaf num-req-rcvd-ero-sent{
    type yang:counter32;
    description
        "The number of requests that have been received for
        which a response with an ERO object was sent. Such
        responses indicate that a path was successfully
        computed by the local PCEP entity.";
}

leaf num-req-rcvd-nopath-sent{
    type yang:counter32;
    description
        "The number of requests that have been received for
        which a response with a NO-PATH object was sent. Such
        responses indicate that the local PCEP entity could
        not find a path to satisfy the request.";
}

leaf num-req-rcvd-cancel-sent{
    type yang:counter32;
    description
        "The number of requests received that were cancelled

by the local PCEP entity sending a PCNtf message. This might be different than num-pcntf-sent because not all PCNtf messages are used to cancel requests, and a single PCNtf message can cancel multiple requests.

leaf num-req-rcvd-error-sent{
  type yang:counter32;
  description
  "The number of requests received that were cancelled by the local PCEP entity sending a PCErr message. This might be different than num-pcerr-sent because not all PCErr messages are used to cancel requests, and a single PCErr message can cancel multiple requests."
}

leaf num-req-rcvd-cancel-rcvd{
  type yang:counter32;
  description
  "The number of requests that were received from the peer and explicitly cancelled by the peer sending a PCNtf."
}

leaf num-rep-rcvd-unknown{
  type yang:counter32;
  description
  "The number of responses to unknown requests received. A response to an unknown request is a response whose RP object does not contain the request ID of any request that is currently outstanding on the session."
}

leaf num-req-rcvd-unknown{
  type yang:counter32;
  description
  "The number of unknown requests that have been received. An unknown request is a request whose RP object contains a request ID of zero."
}

container svec{
  if-feature svec;
  description
  "..."
"If synchronized path computation is supported";
leaf num-svec-sent{
    type yang:counter32;
    description
    "The number of SVEC objects sent in PCReq messages. An SVEC object represents a set of synchronized requests."
}

leaf num-svec-req-sent{
    type yang:counter32;
    description
    "The number of requests sent that appeared in one or more SVEC objects."
}

leaf num-svec-rcvd{
    type yang:counter32;
    description
    "The number of SVEC objects received in PCReq messages. An SVEC object represents a set of synchronized requests."
}

leaf num-svec-req-rcvd{
    type yang:counter32;
    description
    "The number of requests received that appeared in one or more SVEC objects."
}

container stateful{
    if-feature stateful;
    description
    "Stateful PCE related statistics"
    leaf num-pcrpt-sent{
        type yang:counter32;
        description
        "The number of PCRpt messages sent."
    }
    leaf num-pcrpt-rcvd{
        type yang:counter32;
        description
        "The number of PCRpt messages received."
    }
    leaf num-pcupd-sent{

type yang:counter32;
    description
      "The number of PCUpd messages sent."
    }

leaf num-pcupd-rcvd{
    type yang:counter32;
    description
      "The number of PCUpd messages received."
    }

leaf num-rpt-sent{
    type yang:counter32;
    description
      "The number of LSP Reports sent. A LSP report corresponds 1:1 with an LSP object in a PCRpt message. This might be greater than num-pcrpt-sent because multiple reports can be batched into a single PCRpt message."
    }

leaf num-rpt-rcvd{
    type yang:counter32;
    description
      "The number of LSP Reports received. A LSP report corresponds 1:1 with an LSP object in a PCRpt message. This might be greater than num-pcrpt-rcvd because multiple reports can be batched into a single PCRpt message."
    }

leaf num-rpt-rcvd-error-sent{
    type yang:counter32;
    description
      "The number of reports of LSPs received that were responded by the local PCEP entity by sending a PCErr message."
    }

leaf num-upd-sent{
    type yang:counter32;
    description
      "The number of LSP updates sent. A LSP update corresponds 1:1 with an LSP object in a PCUpd message. This might be greater than num-pcupd-sent because multiple updates can be batched into a single PCUpd message."
    }
leaf num-upd-rcvd{
  type yang:counter32;
  description
  "The number of LSP Updates received. A LSP update corresponds 1:1
  with an LSP object in a PCUpd message.
  This might be greater than num-pcupd-rcvd because multiple
  updates can be batched into a single PCUpd message.";
}

leaf num-upd-rcvd-unknown{
  type yang:counter32;
  description
  "The number of updates to unknown LSPs received. An update to an unknown LSP is a
  update whose LSP object does not contain the PLSP-ID of any LSP that is currently
  present.";
}

leaf num-upd-rcvd-undelegated{
  type yang:counter32;
  description
  "The number of updates to not delegated LSPs received. An update to an undelegated LSP is a
  update whose LSP object does not contain the PLSP-ID of any LSP that is currently
  delegated to current PCEP session.";
}

leaf num-upd-rcvd-error-sent{
  type yang:counter32;
  description
  "The number of updates to LSPs received that were responded by the local PCEP entity by sending a
  PCErr message.";
}

container initiation {
  if-feature pce-initiated;
  description
  "PCE-Initiated related statistics";
  leaf num-pcinitiate-sent{
    type yang:counter32;
    description
    "The number of PCInitiate messages sent.";
  }
}
leaf num-pcinitiate-rcvd{
  type yang:counter32;
  description
    "The number of PCInitiate messages received.";
}

leaf num-initiate-sent{
  type yang:counter32;
  description
    "The number of LSP Initiation sent via PCE. A LSP initiation corresponds 1:1 with an LSP object in a PCInitiate message. This might be greater than num-pcinitiate-sent because multiple initiations can be batched into a single PCInitiate message.";
}

leaf num-initiate-rcvd{
  type yang:counter32;
  description
    "The number of LSP Initiation received from PCE. A LSP initiation corresponds 1:1 with an LSP object in a PCInitiate message. This might be greater than num-pcinitiate-rcvd because multiple initiations can be batched into a single PCInitiate message.";
}

leaf num-initiate-rcvd-error-sent{
  type yang:counter32;
  description
    "The number of initiations of LSPs received that were responded by the local PCEP entity by sending a PCErr message.";
}

container path-key {
  if-feature path-key;
  description
    "If Path-Key is supported";
  leaf num-unknown-path-key{
    type yang:counter32;
    description
      "The number of attempts to expand an unknown path-key.";
  }
}
leaf num-exp-path-key{
    type yang:counter32;
    description
        "The number of attempts to expand an expired
        path-key.";
}
leaf num-dup-path-key{
    type yang:counter32;
    description
        "The number of duplicate attempts to expand same
        path-key.";
}
leaf num-path-key-no-attempt{
    type yang:counter32;
    description
        "The number of expired path-keys with no attempt to
        expand it.";
}
//pcep-stats

grouping lsp-state{
    description
        "This grouping defines the attributes for LSP in LSP-DB.
        These are the attributes specifically from the PCEP
        perspective";
    leaf plsp-id{
        type uint32{
            range "1..1048575";
        }
        description
            "A PCEP-specific identifier for the LSP. A PCC
            creates a unique PLSP-ID for each LSP that is
            constant for the lifetime of a PCEP session.
            PLSP-ID is 20 bits with 0 and 0xFFFFF are
            reserved";
    }
    leaf pcc-id{
        type inet:ip-address;
        description
            "The local internet address of the PCC, that
            generated the PLSP-ID.";
    }
    container lsp-ref{
        description
            "reference to ietf-te lsp state";
    }
}
leaf source {
  type leafref {
  }
  description
    "Tunnel sender address extracted from SENDER_TEMPLATE object";
  reference "RFC3209";
}
leaf destination {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:" + "destination";
  }
  description
    "Tunnel endpoint address extracted from SESSION object";
  reference "RFC3209";
}
leaf tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:tunnel-id";
  }
  description
    "Tunnel identifier used in the SESSION that remains constant over the life of the tunnel.";
  reference "RFC3209";
}
leaf lsp-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:lsp-id";
  }
  description
    "Identifier used in the SENDER_TEMPLATE and the FILTER_SPEC that can be changed to allow a sender to share resources with itself.";
  reference "RFC3209";
}
leaf extended-tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:" + "extended-tunnel-id";
  }
  description
    "Extended Tunnel ID of the LSP.";
  reference "RFC3209";
leaf type {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:type";
  }
  description "LSP type P2P or P2MP";
}

leaf admin-state{
  type boolean;
  description
    "The desired operational state";
}

leaf operational-state{
  type operational-state;
  description
    "The operational status of the LSP";
}

container delegated{
  description
    "The delegation related parameters";
  leaf enabled{
    type boolean;
    description
      "LSP is delegated or not";
  }
  leaf pce{
    type leafref {
      path "/pcep-state/entity/peers/peer/addr";
    }
    must "(../enabled = true())"
    { error-message
      "The LSP must be delegated";
      description
        "When LSP is a delegated LSP";
    }
    description
      "The reference to the PCE peer to which LSP is delegated";
  }
  leaf srp-id{
    type uint32;
    description
      "The last SRP-ID-number associated with this LSP.";
  }
}
container initiation {
  if-feature pce-initiated;
  description
    "The PCE initiation related parameters";
  leaf enabled{
    type boolean;
    description
      "LSP is PCE-initiated or not";
  }
  leaf pce{
    type leafref {
      path "/pcep-state/entity/peers/peer/addr";
    }
    must "(./.enabled = true())"
    { error-message
      "The LSP must be PCE-Initiated";
      description
        "When the LSP must be PCE-Initiated";
    }
    description
      "The reference to the PCE
       that initiated this LSP";
  }
  leaf symbolic-path-name{
    type string;
    description
      "The symbolic path name associated with the LSP.";
  }
  leaf last-error{
    type lsp-error;
    description
      "The last error for the LSP.";
  }
  leaf pst{
    type pst;
    default "rsvp-te";
    description
      "The Path Setup Type";
  }
}

}//lsp-state

grouping notification-instance-hdr {
  description

"This group describes common instance specific data for notifications.";

leaf peer-addr {
    type leafref {
        path "/pcep-state/entity/peers/peer/addr";
    }
    description "Reference to peer address";
}

// notification-instance-hdr

grouping notification-session-hdr {
    description "This group describes common session instance specific data for notifications.";
    leaf session-initiator {
        type leafref {
            path "/pcep-state/entity/peers/peer/sessions/" + "session/initiator";
        }
        description "Reference to pcep session initiator leaf";
    }
}

// notification-session-hdr

grouping stateful-pce-parameter {
    description "This group describes stateful PCE specific parameters.";
    leaf state-timeout{
        type uint32;
        units "seconds";
        description "When a PCEP session is terminated, a PCC waits for this time period before flushing LSP state associated with that PCEP session and reverting to operator-defined default parameters or behaviours.";
    }
    leaf redelegation-timeout{
        type uint32;
        units "seconds";
        must "/pcep-state/entity/role = 'pcc'
            or "/pcep-state/entity/role = 'pcc-and-pce'";
    }
}
" and " + 
"{/pcep/entity/capability/stateful/active" + "= true()}"
}  
error-message "The PCEP entity must be PCC";  
description  
 "When PCEP entity is PCC";
}
}

description  
 "When a PCEP session is terminated, a PCC waits for this time period before revoking LSP delegation to a PCE and attempting to redelegate LSPs associated with the terminated PCEP session to an alternate PCE.";
}

leaf rpt-non-pcep-lsp{
  type boolean;
  must "{(pcep-state/entity/role = 'pcc')" + 
  " or " + 
  "{(pcep-state/entity/role = 'pcc-and-pce')"
}  
error-message "The PCEP entity must be PCC";  
description  
 "When PCEP entity is PCC";
}
}

description  
 "If set, a PCC reports LSPs that are not controlled by any PCE (for example, LSPs that are statically configured at the PCC). ";
}

grouping authentication {
  description "Authentication Information";
  choice auth-type-selection {
    description  
    "Options for expressing authentication setting.";
    case auth-key-chain {
      leaf key-chain {
        type key-chain:key-chain-ref;
        description  
        "key-chain name.";
      }
    }
    case auth-key {

leaf key {
    type string;
    description
        "Key string in ASCII format.";
}
container crypto-algorithm {
    uses key-chain:crypto-algorithm-types;
    description
        "Cryptographic algorithm associated
         with key."
    }
}
case auth-tls {
    if-feature tls;
    container tls {
        description
            "TLS related information - TBD"
    }
}
}

grouping path-key {
    description "Path-key related information";
    leaf enabled{
        type boolean;
        description
            "Enabled or Disabled"
    }
    leaf discard-timer {
        type uint32;
        units "minutes";
        default 10;
        description
            "A timer to discard unwanted path-keys"
    }
    leaf reuse-time {
        type uint32;
        units "minutes";
        default 30;
        description
            "A time after which the path-keys could be reused"
    }
    leaf pce-id {
        type inet:ip-address;
        description
            "PCE Address to be used in each Path-Key Subobject
            (PKS)"
    }
}
grouping path-key-state {
  description "Table to allow inspection of path-keys";
  list path-keys{
    key "path-key";
    description "The list of path-keys generated by the PCE";
    leaf path-key {
      type uint16;
      description "The identifier, or token used to represent
                   the Confidential Path Segment (CPS) within
                   the context of the PCE";
    }
  }
  container cps {
    description "The Confidential Path Segment (CPS)";
    list explicit-route-objects {
      key "index";
      description "List of explicit route objects";
      leaf index {
        type uint8 {
          range "0..255";
        }
        description "Index of this explicit route object";
      }
      leaf explicit-route-usage {
        type identityref {
          base te-types:route-usage-type;
        }
        description "An explicit-route hop action.";
      }
      uses te-types:explicit-route-subobject;
    }
    leaf pcc-original {
      type leafref {
        path "/pcep-state/entity/peers/peer/addr";
      }
      description "Reference to PCC peer address of";
    }
  }
}

leaf req-id {
    type uint32;
    description
        "The request ID of the original PCReq.";
}
leaf retrieved {
    type boolean;
    description
        "If path-key has been retrieved yet";
}
leaf pcc-retrieved {
    type leafref {
        path "/pcep-state/entity/peers/peer/addr";
    } must "(../retrieved = true())"
    { error-message
        "The Path-key should be retrieved";
        description
            "When Path-Key has been retrieved";
    }
    description
        "Reference to PCC peer address which retrieved the path-key";
}
leaf creation-time {
    type yang:timestamp;
    description
        "The timestamp value at the time this Path-Key was created.";
}
leaf discard-time {
    type uint32;
    units "minutes";
    description
        "A time after which this path-keys will be discarded";
}
leaf reuse-time {
    type uint32;
    units "minutes";
    description
        "A time after which this path-keys could be reused";
}
grouping of-list {
  description "List of OF";
  list objective-function{
    key "of";
    description "The list of authorized OF";
    leaf of {
      type objective-function;
      description "The OF authorized";
    }
  }
}

grouping association {
  description "Generic Association parameters";
  leaf type {
    type "assoc-type";
    description "The PCEP association type";
  }
  leaf id {
    type uint16;
    description "PCEP Association ID";
  }
  leaf source {
    type inet:ip-address;
    description "PCEP Association Source.";
  }
  leaf global-source {
    type uint32;
    description "PCEP Association Global Source.";
  }
  leaf extended-id{
    type string;
    description "Additional information to support unique identification.";
  }
}
grouping association-ref {
    description "Generic Association parameters";
    leaf id {
        type leafref {
            path "/pcep-state/entity/lsp-db/association-list/id";
        }
        description "PCEP Association ID";
    }
    leaf source {
        type leafref {
            path "/pcep-state/entity/lsp-db/association-list/source";
        }
        description "PCEP Association Source.";
    }
    leaf global-source {
        type leafref {
            path "/pcep-state/entity/lsp-db/association-list/global-source";
        }
        description "PCEP Association Global Source.";
    }
    leaf extended-id {
        type leafref {
            path "/pcep-state/entity/lsp-db/association-list/extended-id";
        }
        description "Additional information to support unique identification.";
    }
}

/*
 * Configuration data nodes
 */
container pcep{
    presence "The PCEP is enabled";
    description "Parameters for list of configured PCEP entities"
on the device.

container entity {

description "The configured PCEP entity on the device.";

leaf addr {
    type inet:ip-address;
    mandatory true;
    description "The local Internet address of this PCEP entity.
    If operating as a PCE server, the PCEP entity listens on this address.
    If operating as a PCC, the PCEP entity binds outgoing TCP connections to this address.
    It is possible for the PCEP entity to operate both as a PCC and a PCE Server, in which case it uses this address both to listen for incoming TCP connections and to bind outgoing TCP connections.";
}

leaf enabled {
    type boolean;
    default true;
    description "The administrative status of this PCEP Entity.";
}

leaf role {
    type pcep-role;
    mandatory true;
    description "The role that this entity can play.
    Takes one of the following values.
    - unknown(0): this PCEP Entity role is not known.
    - pcc(1): this PCEP Entity is a PCC.
    - pce(2): this PCEP Entity is a PCE.
    - pcc-and-pce(3): this PCEP Entity is both a PCC and a PCE.";
}

leaf description {


type string;
description
   "Description of the PCEP entity configured
    by the user";
}

leaf speaker-entity-id{
   if-feature stateful-sync-opt;
   type string;
   description
      "The Speaker Entity Identifier";
}

uses info {
   description
      "Local PCEP entity information";
}

container pce-info {
   must "((/pcep-state/entity/role = 'pce')" +
      " or " +
      "(/pcep-state/entity/role = 'pcc-and-pce'))"
   {  
      error-message "The PCEP entity must be PCE";
      description
         "When PCEP entity is PCE";
      }
   uses pce-info {
      description
         "Local PCE information";
   }
}

container path-key {
   if-feature path-key;
   uses path-key {
      description
         "Path-Key Configuration";
   }
   description
         "Path-Key Configuration";
}

description
   "The Local PCE Entity PCE information";
}

uses authentication {
   description

"Local PCEP entity authentication information";
}

uses pcep-entity-info {
    description
    "The configuration related to the PCEP entity.";
}

leaf pcep-notification-max-rate {
    type uint32;
    mandatory true;
    description
    "This variable indicates the maximum number of notifications issued per second. If events occur more rapidly, the implementation may simply fail to emit these notifications during that period, or may queue them until an appropriate time. A value of 0 means no notifications are emitted and all should be discarded (that is, not queued).";
}

container stateful-parameter{
    if-feature stateful;
    must "(/pcep/entity/capability/stateful/enabled" + " = true())"
    {
        error-message
        "The Stateful PCE must be enabled";
        description
        "When PCEP entity is stateful enabled";
    }
    uses stateful-pce-parameter;
    description
    "The configured stateful parameters";
}

container of-list{
    if-feature objective-function;
    must "((/pcep/entity/role = ‘pce’)" + " or " + "(/pcep/entity/role = ‘pcc-and-pce’))"
{ error-message
    "The PCEP entity must be PCE";
    description
        "The authorized OF-List at PCE";
}
uses of-list;

description
    "The authorized OF-List at PCE for all peers";
}

container peers{
    must "((/pcep/entity/role = 'pcc')" + 
    " or " + 
    "(/pcep/entity/role = 'pcc-and-pce'))"
    { error-message
        "The PCEP entity must be PCC";
        description
            "When PCEP entity is PCC, as remote
            PCE peers are configured.";
    }
    description
        "The list of configured peers for the
        entity (remote PCE)";
    list peer{
        key "addr";
        description
            "The peer configured for the entity. 
            (remote PCE)";
        leaf addr {
            type inet:ip-address;
            description
                "The local Internet address of this 
                PCEP peer.";
        }
        leaf description {
            type string;
            description
                "Description of the PCEP peer 
                configured by the user";
        }
        uses info {

description
  "PCE Peer information";
}
uses pce-info {
  description
  "PCE Peer information";
}
leaf delegation-pref{
  if-feature stateful;
  type uint8{
    range "0..7";
  }
  must "((/pcep/entity/capability/stateful/active" + "= true()))"
  {
    error-message
    "The Active Stateful PCE must be enabled";
    description
    "When PCEP entity is active stateful enabled";
  }
  description
  "The PCE peer delegation preference.";
}
uses authentication {
  description
  "PCE Peer authentication";
}
container of-list{
  if-feature objective-function;
  must "((/pcep/entity/role = 'pce')" + " or " + "(/pcep/entity/role = 'pcc-and-pce'))"
  {
    error-message
    "The PCEP entity must be PCE";
    description
    "The authorized OF-List at PCE";
  }
  uses of-list;
  description
  "The authorized OF-List a specific peer";
}
}//peer
}//peers
container pcep-state{
  config false;
  description "The list of operational PCEP entities on the device.";
}

container entity{
  description "The operational PCEP entity on the device.";
  leaf addr {
    type inet:ip-address;
    description "The local Internet address of this PCEP entity. If operating as a PCE server, the PCEP entity listens on this address. If operating as a PCC, the PCEP entity binds outgoing TCP connections to this address. It is possible for the PCEP entity to operate both as a PCC and a PCE Server, in which case it uses this address both to listen for incoming TCP connections and to bind outgoing TCP connections.";
  }
  leaf index{
    type uint32;
    description "The index of the operational PCEP entity";
  }
  leaf admin-status {
    type pcep-admin-status;
    description "The administrative status of this PCEP Entity. This is the desired operational status as currently set by an operator or by default in
the implementation. The value of enabled represents the current status of an attempt to reach this desired status."

leaf oper-status {
    type pcep-admin-status;
    description
        "The operational status of the PCEP entity. Takes one of the following values.
        - oper-status-up(1): the PCEP entity is active.
        - oper-status-down(2): the PCEP entity is inactive.
        - oper-status-going-up(3): the PCEP entity is activating.
        - oper-status-going-down(4): the PCEP entity is deactivating.
        - oper-status-failed(5): the PCEP entity has failed and will recover when possible.
        - oper-status-failed-perm(6): the PCEP entity has failed and will not recover without operator intervention.";
}

leaf role {
    type pcep-role;
    description
        "The role that this entity can play. Takes one of the following values.
        - unknown(0): this PCEP entity role is not known.
        - pcc(1): this PCEP entity is a PCC.
        - pce(2): this PCEP entity is a PCE.
        - pcc-and-pce(3): this PCEP entity is both a PCC and a PCE."
}

leaf description {
    type string;
    description
        "Description of the PCEP entity configured by the user";
}

leaf speaker-entity-id{
    if-feature stateful-sync-opt;
    type string;
}
uses info {
    description
    "Local PCEP entity information";
}

container pce-info {
    when "((/pcep-state/entity/role = 'pce')" +
    " or "+
    "((/pcep-state/entity/role = 'pcc-and-pce'))"
    {
        description
        "When PCEP entity is PCE";
    }
    uses pce-info {
        description
        "Local PCE information";
    }
}

container path-key {
    if-feature path-key;
    uses path-key {
        description
        "Path-Key Configuration";
    }
    description
    "Path-Key Configuration";
}

description
    "The Local PCE Entity PCE information";
}

uses authentication {
    description
    "Local PCEP Entity authentication information";
}

uses pcep-entity-info{
    description
    "The operational information related to the
    PCEP entity.";
}

container stateful-parameter{

if-feature stateful;
must "({pcep/entity/capability/stateful/enabled} + " = true())"
{
  error-message
  "The Stateful PCE must be enabled";
  description
  "When PCEP entity is stateful enabled";
}
uses stateful-pce-parameter;
description
  "The operational stateful parameters";
}

container lsp-db{
  if-feature stateful;
description
  "The LSP-DB";
leaf db-ver{
  if-feature stateful-sync-opt;
type uint64;
must "({pcep/entity/role = 'pcc')" + " or " + "({pcep/entity/role = 'pcc-and-pce'))" 
{
  error-message
    "The PCEP entity must be PCC";
  description
    "When PCEP entity is PCC, as remote PCE peers are configured.";
}
description
  "The LSP State Database Version Number";
}
list association-list {
  key "id source global-source extended-id";
description
  "List of all PCEP associations";
  uses association {
    description
      "The Association attributes";
  }
list lsp {

key "plsp-id pcc-id";
description
  "List of all LSP in this association";
leaf plsp-id {
  type leafref {
    path "/pcep-state/entity/lsp-db/" + "lsp/plsp-id";
  }
description
  "Reference to PLSP-ID in LSP-DB";
}
leaf pcc-id {
  type leafref {
    path "/pcep-state/entity/lsp-db/" + "lsp/pcc-id";
  }
description
  "Reference to PCC-ID in LSP-DB";
}
list lsp{
  key "plsp-id pcc-id";
  description
  "List of all LSPs in LSP-DB";
  uses lsp-state{
    description
    "The PCEP specific attributes for LSP-DB.";
  }
}
list association-list {
  key "id source global-source extended-id";
  description
  "List of all PCEP associations";
  uses association-ref {
    description
    "Reference to the Association attributes";
  }
}
}
}
container path-keys {
  if-feature path-key;
  must "((/pcep-state/entity/role = 'pce')" + " or " + "(/pcep-state/entity/role = 'pcc-and-pce')")"


```yang
{  error-message
    "The PCEP entity must be PCE";
  description
    "When PCEP entity is PCE";
}
uses path-key-state;
description
  "The path-keys generated by the PCE";
}
container of-list{
  if-feature objective-function;
  must "((/pcep/entity/role = 'pce')" +
  " or " +
  "(/pcep/entity/role = 'pcc-and-pce'))"
  error-message
    "The PCEP entity must be PCE";
  description
    "The authorized OF-List at PCE";
}
uses of-list;
description
  "The authorized OF-List at PCE for all peers";
}
container peers{
  description
    "The list of peers for the entity";
  list peer{
    key "addr";
    description
      "The peer for the entity.";
    leaf addr {
      type inet:ip-address;
      description
        "The local Internet address of this PCEP peer.";
    }
    leaf role {
      type pcep-role;
      description
        "The role of the PCEP Peer.
        Takes one of the following values.
        "} 
```
- unknown(0): this PCEP peer role
  is not known.
- pcc(1): this PCEP peer is a PCC.
- pce(2): this PCEP peer is a PCE.
- pcc-and-pce(3): this PCEP peer
  is both a PCC and a PCE.

uses info {
  description
  "PCEP peer information";
}

container pce-info {
  when "((/pcep-state/entity/role = 'pcc')" +
  " or " +
  "(/pcep-state/entity/role = 'pcc-and-pce'))"
  {
    description
    "When PCEP entity is PCE";
  }
  uses pce-info {
    description
    "PCE Peer information";
  }
  description
  "The PCE Peer information";
}

leaf delegation-pref{
  if-feature stateful;
  type uint8{
    range "0..7";
  }
  must "((/pcep-state/entity/role = 'pcc')" +
  " or " +
  "(/pcep-state/entity/role = 'pcc-and-pce'))"
  {
    error-message
    "The PCEP entity must be PCC";
    description
    "When PCEP entity is PCC";
  }
  must "(/pcep/entity/capability/stateful/active"
+ "= true()"
{
  error-message
  "The Active Stateful PCE must be enabled";
  description
  "When PCEP entity is active stateful enabled";
}

description
"The PCE peer delegation preference.";
}

uses authentication {
  description
  "PCE Peer authentication";
}

container of-list{
  if-feature objective-function;
  must "((/pcep/entity/role = 'pce')" + " or " + "(/pcep/entity/role = 'pcc-and-pce'))"
{
  error-message
  "The PCEP entity must be PCE";
  description
  "The authorized OF-List at PCE";
}

uses of-list;

description
"The authorized OF-List of a specific peer";
}

leaf discontinuity-time {
  type yang:timestamp;
  description
  "The timestamp of the time when the information and statistics were last reset.";
}

leaf initiate-session {
  type boolean;
  description
  "Indicates whether the local PCEP entity initiates sessions to this peer,
or waits for the peer to initiate a session.
}

leaf session-exists{
  type boolean;
  description
    "Indicates whether a session with this peer currently exists."
}

leaf num-sess-setup-ok{
  type yang:counter32;
  description
    "The number of PCEP sessions successfully established with the peer, including any current session. This counter is incremented each time a session with this peer is successfully established."
}

leaf num-sess-setup-fail{
  type yang:counter32;
  description
    "The number of PCEP sessions with the peer that have been attempted but failed before being fully established. This counter is incremented each time a session retry to this peer fails."
}

leaf session-up-time{
  type yang:timestamp;
  must "/num-sess-setup-ok != 0 or " + "/num-sess-setup-ok = 0 and " + "/(= 0))" {
    error-message
      "Invalid Session Up timestamp";
    description
      "If num-sess-setup-ok is zero, then this leaf contains zero."
  }
  description
    "The timestamp value of the last time a session with this peer was successfully established.";
}
leaf session-fail-time{
  type yang:timestamp;
  must "(./.num-sess-setup-fail != 0 or " +
   "(./.num-sess-setup-fail = 0 and " +
   "(. = 0)))" {
    error-message
    "Invalid Session Fail timestamp";
    description
    "If num-sess-setup-fail is zero,
     then this leaf contains zero.";
  }
  description
  "The timestamp value of the last time a
   session with this peer failed to be
   established.";
}

leaf session-fail-up-time{
  type yang:timestamp;
  must "(./.num-sess-setup-ok != 0 or " +
   "(./.num-sess-setup-ok = 0 and " +
   "(. = 0)))" {
    error-message
    "Invalid Session Fail from
     Up timestamp";
    description
    "If num-sess-setup-ok is zero,
     then this leaf contains zero.";
  }
  description
  "The timestamp value of the last time a
   session with this peer failed from
   active.";
}

container pcep-stats {
  description
  "The container for all statistics at peer
   level.";
  uses pcep-stats{
    description
    "Since PCEP sessions can be
     ephemeral, the peer statistics tracks
     a peer even when no PCEP session
     currently exists to that peer. The
     statistics contained are an aggregate
     of the statistics for all successive
     sessions to that peer.";
}
leaf num-req-sent-closed{
    type yang:counter32;
    description
        "The number of requests that were
        sent to the peer and implicitly
        cancelled when the session they were
        sent over was closed.";
}

leaf num-req-rcvd-closed{
    type yang:counter32;
    description
        "The number of requests that were
        received from the peer and
        implicitly cancelled when the
        session they were received over
        was closed.";
}

} //pcep-stats

container sessions {
    description
        "This entry represents a single PCEP
        session in which the local PCEP entity
        participates.  This entry exists only if the
        corresponding PCEP session has been
        initialized by some event, such as
        manual user configuration, auto-
        discovery of a peer, or an incoming
        TCP connection.";

description
        "The list of sessions, note that
        for a time being two sessions
        may exist for a peer";

    list session {
        key "initiator";

        description
            "The initiator of the session,";

        leaf initiator {
            type pcep-initiator;
            description
                "The initiator of the session,";

    } //session

} //pcep-sessions
that is, whether the TCP connection was initiated by the local PCEP entity or the peer.
There is a window during session initialization where two sessions can exist between a pair of PCEP speakers, each initiated by one of the speakers. One of these sessions is always discarded before it leaves OpenWait state. However, before it is discarded, two sessions to the given peer appear transiently in this MIB module. The sessions are distinguished by who initiated them, and so this field is the key.

 leaf state-last-change {
   type yang:timestamp;
   description
   "The timestamp value at the time this session entered its current state as denoted by the state leaf.";
 }

 leaf state {
   type pcep-sess-state;
   description
   "The current state of the session. The set of possible states excludes the idle state since entries do not exist in the idle state.";
 }

 leaf session-creation {
   type yang:timestamp;
   description
   "The timestamp value at the time this session was created.";
 }

leaf connect-retry {
  type yang:counter32;
  description
  "The number of times that the
  local PCEP entity has
  attempted to establish a TCP
  connection for this session
  without success. The PCEP
  entity gives up when this
  reaches connect-max-retry.";
}

leaf local-id {
  type uint32 {
    range "0..255";
  }
  description
  "The value of the PCEP session
  ID used by the local PCEP
  entity in the Open message
  for this session.
  If state is tcp-pending then
  this is the session ID that
  will be used in the Open
  message. Otherwise, this is
  the session ID that was sent
  in the Open message.";
}

leaf remote-id {
  type uint32 {
    range "0..255";
  }
  must "((../state != 'tcp-pending')
   "and " +
   "../state != 'open-wait')" +
   "or " +
   "((../state = 'tcp-pending')
   "or " +
   "../state = 'open-wait')" +
   "and (.. = 0))"
    {
      error-message
      "Invalid remote-id";
      description
      "If state is tcp-pending
      or open-wait then this
      leaf is not used and
      MUST be set to zero.";
    }
leaf keepalive-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  must "(. /= State = 'session-up')" + 
      "or" + 
      "(. /= State != 'session-up')" + 
      "and (. = 0))" {
    error-message
    "Invalid keepalive timer";
    description
    "This field is used if and only if state is session-up. Otherwise, it is not used and MUST be set to zero.";
  }

  description
  "The agreed maximum interval at which the local PCEP entity transmits PCEP messages on this PCEP session. Zero means that the local PCEP entity never sends Keepalives on this session.";
}

leaf peer-keepalive-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  must "(. /= State = 'session-up')" + 
      "or" + 
      "(. /= State != 'session-up')" + 
      "and (. = 0))" {

error-message
  "Invalid Peer keepalive timer";
description
  "This field is used if and only if state is session-up. Otherwise, it is not used and MUST be set to zero."
}
description
  "The agreed maximum interval at which the peer transmits PCEP messages on this PCEP session. Zero means that the peer never sends Keepalives on this session."
}
leaf dead-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  description
    "The dead timer interval for this PCEP session."
}
leaf peer-dead-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  must "((../state != 'tcp-pending') + "and " + "../state != 'open-wait')" + "or " + "((../state = 'tcp-pending') + "or " + "../state = 'open-wait')" + "and " + "(. = 0))"
    error-message
      "Invalid Peer Dead timer";
    description
      "If state is tcp-
pending or open-wait
then this leaf is not
used and MUST be set to
zero."
}
description
"The peer’s dead-timer interval
for this PCEP session.";
}

leaf ka-hold-time-rem {
type uint32 {
  range "0..255";
}
units "seconds";
must "((../state != 'tcp-pending' +
  "and " +
  "./.state != 'open-wait' ) " +
  "or " +
  "((../state = 'tcp-pending' +
    "or " +
    "./.state = 'open-wait' )" +
    "and " +
    "(. = 0))\)" {
  error-message
  "Invalid Keepalive hold
time remaining";
  description
  "If state is tcp-pending
  or open-wait then this
  field is not used and
  MUST be set to zero.";
}
description
"The keep alive hold time
remaining for this session.";
}

leaf overloaded {
type boolean;
description
"If the local PCEP entity has
informed the peer that it is
currently overloaded, then this
field is set to true. Otherwise, it
is set to false.";
}
leaf overload-time {
  type uint32;
  units "seconds";
  must "((../overloaded = true()) or" +
    "((../overloaded != true()) and" +
    " (. = 0)))" {
    error-message
    "Invalid overload-time";
    description
    "This field is only used
    if overloaded is set to
    true. Otherwise, it is
    not used and MUST be set
    to zero.";
  }
  description
  "The interval of time that is
  remaining until the local PCEP
  entity will cease to be
  overloaded on this session.";
}

leaf peer-overloaded {
  type boolean;
  description
  "If the peer has informed the
  local PCEP entity that it is
  currently overloaded, then this
  is set to true. Otherwise, it
  is set to false.";
}

leaf peer-overload-time {
  type uint32;
  units "seconds";
  must "((../peer-overloaded = true())" +
    " or " +
    "((../peer-overloaded != true())" +
    " and " +
    " (. = 0)))" {
    error-message
    "Invalid peer overload
    time";
    description
    "This field is only used
    if peer-overloaded is
    set to true. Otherwise,
    it is not used and MUST
be set to zero.
}

description
"The interval of time that is
remaining until the peer will
cease to be overloaded. If it
is not known how long the peer
will stay in overloaded state,
this leaf is set to zero.";

leaf lspdb-sync {
  if-feature stateful;
type sync-state;
description
  "The LSP-DB state synchronization
  status."
}

leaf recv-db-ver{
  if-feature stateful;
  if-feature stateful-sync-opt;
type uint64;
must "((/pcep-state/entity/peers/" +
  "peer/role = 'pcc')" +
  " or " +
  "((/pcep-state/entity/peers/" +
  "peer/role = 'pcc-and-pce'))"
{
  error-message
  "The PCEP peer must be PCC";
description
  "The PCEP peer must be PCC";
}
description
"The last received LSP State
Database Version Number";

}

container of-list{
  if-feature objective-function;
must "((/pcep/entity/role = 'pcc')" +
  " or " +
  "((/pcep/entity/role = 'pcc-and-pce'))"
{
  error-message
  "The PCEP entity must be PCC";
description
  "The OF-list received on the

session";
}
uses of-list;
description
 "Indicate the list of supported OF
 on this session";
}

leaf speaker-entity-id{
   if-feature stateful-sync-opt;
type string;
description
 "The Speaker Entity Identifier";
}

leaf discontinuity-time {
type yang:timestamp;
description
 "The timestamp value of the time
 when the statistics were last
 reset.";
}

container pcep-stats {
description
 "The container for all statistics
 at session level.";
uses pcep-stats{
description
 "The statistics contained are
 for the current sessions to
 that peer. These are lost
 when the session goes down."

}
}//pcep-stats
} // session
} // sessions
} // peer
} // peers
} // entity
} // pcep-state

/*
 * Notifications
 */
notification pcep-session-up {
  description
    "This notification is sent when the value of
    '/pcep/pcep-state/peers/peer/sessions/session/state'
    enters the 'session-up' state."

  uses notification-instance-hdr;

  uses notification-session-hdr;

  leaf state-last-change {
    type yang:timestamp;
    description
      "The timestamp value at the time this session entered
       its current state as denoted by the state leaf."
  }

  leaf state {
    type pcep-sess-state;
    description
      "The current state of the session.
       The set of possible states excludes the idle state
       since entries do not exist in the idle state.";
  }
}

notification pcep-session-down {
  description
    "This notification is sent when the value of
    '/pcep/pcep-state/peers/peer/sessions/session/state'
    leaves the 'session-up' state."

  uses notification-instance-hdr;

  leaf session-initiator {
    type pcep-initiator;
    description
      "The initiator of the session.";
  }

  leaf state-last-change {
    type yang:timestamp;
    description
      "The timestamp value at the time this session entered
       its current state as denoted by the state leaf."
  }

  leaf state {

type pcep-sess-state;

description
"The current state of the session. The set of possible states excludes the idle state since entries do not exist in the idle state."

} //notification

notification pcep-session-local-overload {

description
"This notification is sent when the local PCEP entity enters overload state for a peer."

uses notification-instance-hdr;

uses notification-session-hdr;

leaf overloaded {

type boolean;

description
"If the local PCEP entity has informed the peer that it is currently overloaded, then this is set to true. Otherwise, it is set to false."

}

leaf overload-time {

type uint32;

units "seconds";

description
"The interval of time that is remaining until the local PCEP entity will cease to be overloaded on this session."

}

} //notification

notification pcep-session-local-overload-clear {

description
"This notification is sent when the local PCEP entity leaves overload state for a peer."

uses notification-instance-hdr;

leaf overloaded {

type boolean;

description
"If the local PCEP entity has informed the peer that it is currently overloaded, then this is set to true. Otherwise, it is set to false."

}
notification pcep-session-peer-overload {
    description
    "This notification is sent when a peer enters overload state."

    uses notification-instance-hdr;
    uses notification-session-hdr;

    leaf peer-overloaded {
        type boolean;
        description
        "If the peer has informed the local PCEP entity that it is currently overloaded, then this is set to true. Otherwise, it is set to false.";
    }

    leaf peer-overload-time {
        type uint32;
        units "seconds";
        description
        "The interval of time that is remaining until the peer will cease to be overloaded. If it is not known how long the peer will stay in overloaded state, this leaf is set to zero.";
    }
}

notification pcep-session-peer-overload-clear {
    description
    "This notification is sent when a peer leaves overload state."

    uses notification-instance-hdr;

    leaf peer-overloaded {
        type boolean;
        description
        "If the peer has informed the local PCEP entity that it is currently overloaded, then this is set to true. Otherwise, it is set to false.";
    }
} //notification
} //module
9. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

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

TBD: List specific Subtrees and data nodes and their sensitivity/vulnerability.

10. Manageability Considerations

10.1. Control of Function and Policy

10.2. Information and Data Models

10.3. Liveness Detection and Monitoring

10.4. Verify Correct Operations

10.5. Requirements On Other Protocols

10.6. Impact On Network Operations

11. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.


Registrant Contact: The PCE WG of the IETF.

XML: N/A; the requested URI is an XML namespace.
This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

Name:         ietf-pcep
Prefix:       pcep
Reference:    This I-D

12. Acknowledgements

The initial document is based on the PCEP MIB [RFC7420]. Further this document structure is based on Routing Yang Module [I-D.ietf-netmod-routing-cfg]. We would like to thank the authors of aforementioned documents.

13. References

13.1. Normative References


[I-D.ietf-pce-stateful-pce]
13.2. Informative References


Appendix A. Contributor Addresses

Rohit Pobbathi
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
EMail: rohit.pobbathi@huawei.com

Vinod KumarS
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
EMail: vinods.kumar@huawei.com

Zafar Ali
Cisco Systems
Canada
EMail: zali@cisco.com

Xufeng Liu
Ericsson
1595 Spring Hill Road, Suite 500
Vienna, VA 22182
USA
EMail: xufeng.liu@ericsson.com

Young Lee
Huawei Technologies
5340 Legacy Drive, Building 3
Plano, TX 75023, USA
Phone: (469) 277-5838
EMail: leeyoung@huawei.com

Udayasree Palle
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
EMail: udayasree.palle@huawei.com

A YANG Data Model for Path Computation Element Communications Protocol (PCEP)
draft-ietf-pce-pcep-yang-12

Abstract

This document defines a YANG data model for the management of Path Computation Element communications Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs. The data model includes configuration and state data.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 2, 2020.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents before using this document.
Internet-Draft          PCE-YANG       July 2019

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.  Introduction .............................................. 3
2.  Requirements Language ................................. 3
3.  Terminology and Notation ............................... 3
   3.1.  Tree Diagrams .................................. 5
   3.2.  Prefixes in Data Node Names ..................... 5
   3.3.  References in the Model ......................... 5
4.  Objectives ............................................... 7
5.  The Design of PCEP Data Model ......................... 8
   5.1.  The Overview of PCEP Data Model ................. 8
   5.2.  The Entity ...................................... 9
   5.3.  The Peer Lists .................................. 13
   5.4.  The Session Lists ................................ 15
   5.5.  Notifications ................................... 17
   5.6.  RPC ............................................. 17
   5.7.  The Full PCEP Data Model ....................... 18
6.  The Design of PCEP Statistics Data Model ............ 24
7.  Advanced PCE Features .................................. 27
   7.1.  Stateful PCE’s LSP-DB ........................... 28
8.  Open Issues and Next Step .............................. 28
   8.1.  The PCE-Initiated LSP ........................... 28
9.  Other Considerations .................................... 28
   9.1.  PCEP over TLS (PCEPS) ............................ 28
10. PCEP YANG Modules ...................................... 28
   10.1. ietf-pcep module ................................ 29
   10.2. ietf-pcep-stats module ......................... 83
11. Security Considerations ................................. 101
12. IANA Considerations ..................................... 102
13. Implementation Status .................................... 103
14. Acknowledgements ........................................ 103
15. References ............................................... 104
   15.1. Normative References ................................ 104
   15.2. Informative References ............................ 107
Appendix A.  Example .......................................... 109
Appendix B.  Contributor Addresses .......................... 112
Authors’ Addresses ........................................... 113
1. Introduction

The Path Computation Element (PCE) defined in [RFC4655] is an entity that is capable of computing a network path or route based on a network graph, and applying computational constraints. A Path Computation Client (PCC) may make requests to a PCE for paths to be computed.

PCEP is the communication protocol between a PCC and PCE and is defined in [RFC5440]. PCEP interactions include path computation requests and path computation replies as well as notifications of specific states related to the use of a PCE in the context of Multiprotocol Label Switching (MPLS) and Generalized MPLS (GMPLS) Traffic Engineering (TE). [RFC8231] specifies extensions to PCEP to enable stateful control of MPLS TE LSPs.

This document defines a YANG [RFC7950] data model for the management of PCEP speakers. It is important to establish a common data model for how PCEP speakers are identified, configured, and monitored. The data model includes configuration data and state data.

This document contains a specification of the PCEP YANG module, "ietf-pcep" which provides the PCEP [RFC5440] data model.

The PCEP operational state is included in the same tree as the PCEP configuration consistent with Network Management Datastore Architecture [RFC8342]. The origin of the data is indicated as per the origin metadata annotation.

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.

3. Terminology and Notation

This document uses the terminology defined in [RFC4655] and [RFC5440]. In particular, it uses the following acronyms.

- Path Computation Request message (PCReq).
- Path Computation Reply message (PCRep).
- Notification message (PCNtf).
o Error message (PCErr).
o Request Parameters object (RP).
o Synchronization Vector object (SVEC).
o Explicit Route object (ERO).

This document also uses the following terms defined in [RFC7420]:

o PCEP entity: a local PCEP speaker.
o PCEP peer: to refer to a remote PCEP speaker.
o PCEP speaker: where it is not necessary to distinguish between local and remote.

Further, this document also uses the following terms defined in [RFC8231]:

o Stateful PCE, Passive Stateful PCE, Active Stateful PCE

o Delegation, Revocation, Redelegation

o LSP State Report, Path Computation Report message (PCRpt).

o LSP State Update, Path Computation Update message (PCUpd).

[RFC8281]:

o PCE-initiated LSP, Path Computation LSP Initiate Message (PCInitiate).

[RFC8408]:

o Path Setup Type (PST).

[I-D.ietf-pce-segment-routing]:

o Segment Routing (SR).

[RFC6241]:

o Configuration data.

o State data.
3.1. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is defined in [RFC8340].

3.2. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>te</td>
<td>ietf-te</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[I-D.ietf-teas-yang-te]</td>
</tr>
<tr>
<td>key-chain</td>
<td>ietf-key-chain</td>
<td>[RFC8177]</td>
</tr>
<tr>
<td>nacm</td>
<td>ietf-netconf-acm</td>
<td>[RFC8341]</td>
</tr>
<tr>
<td>tlss</td>
<td>ietf-tls-server</td>
<td>[I-D.ietf-netconf-tls-client-server]</td>
</tr>
<tr>
<td>tlsc</td>
<td>ietf-tls-client</td>
<td>[I-D.ietf-netconf-tls-client-server]</td>
</tr>
<tr>
<td>ospf</td>
<td>ietf-ospf</td>
<td>[I-D.ietf-ospf-yang]</td>
</tr>
<tr>
<td>isis</td>
<td>ietf-isis</td>
<td>[I-D.ietf-isis-yang-isis-cfg]</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

3.3. References in the Model

Following documents are referenced in the model defined in this document -

<table>
<thead>
<tr>
<th>Documents</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSVP-TE: Extensions to LSP Tunnels</td>
<td>[RFC3209]</td>
</tr>
<tr>
<td>RSVP for LSP Tunnels</td>
<td>[RFC5088]</td>
</tr>
<tr>
<td>OSPF Protocol Extensions for Path Computation</td>
<td>[RFC5089]</td>
</tr>
</tbody>
</table>

Dhody, et al. Expires January 2, 2020
<table>
<thead>
<tr>
<th>Path Computation Element (PCE) Discovery</th>
<th>[RFC5440]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Computation Element (PCE) Communication Protocol (PCEP)</td>
<td>[RFC5520]</td>
</tr>
<tr>
<td>Preserving Topology</td>
<td>[RFC5541]</td>
</tr>
<tr>
<td>Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism</td>
<td>[RFC5557]</td>
</tr>
<tr>
<td>Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)</td>
<td>[RFC5571]</td>
</tr>
<tr>
<td>Common YANG Data Types</td>
<td>[RFC8177]</td>
</tr>
<tr>
<td>YANG Data Model for Key Chains</td>
<td>[RFC8231]</td>
</tr>
<tr>
<td>Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE Optimizations of Label Switched Path State</td>
<td>[RFC8232]</td>
</tr>
<tr>
<td>Synchronization Procedures for a Stateful PCE</td>
<td>[RFC8253]</td>
</tr>
<tr>
<td>PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)</td>
<td>[RFC8281]</td>
</tr>
<tr>
<td>Path Computation Element Communication Protocol (PCEP) Extensions for PCE-Initiated LSP Setup in a Stateful PCE Model Extensions to the Path Computation Element</td>
<td>[RFC8306]</td>
</tr>
<tr>
<td>Engineering Label Switched</td>
<td>[RFC8306]</td>
</tr>
</tbody>
</table>
4. Objectives

This section describes some of the design objectives for the model:

- In case of existing implementations, it needs to map the data model defined in this document to their proprietary native data model. To facilitate such mappings, the data model should be simple.

- The data model should be suitable for new implementations to use as is.

- Mapping to the PCEP MIB Module should be clear.

- The data model should allow for static configurations of peers.

- The data model should include read-only counters in order to gather statistics for sent and received PCEP messages, received messages with errors, and messages that could not be sent due to errors. This could be in a separate model which augments the base data model.
It should be fairly straightforward to augment the base data model for advanced PCE features.

5. The Design of PCEP Data Model

5.1. The Overview of PCEP Data Model

The PCEP YANG module defined in this document has all the common building blocks for the PCEP protocol.

module: ietf-pcep
   +--rw pcep!
     +--rw entity
      |   +--rw addr inet:ip-address
      |   +--rw enabled? boolean
      |   +--rw role pcep-role
      |   +--rw description? string
      |   |   (stateful-sync-opt)?
      |   +--rw speaker-entity-id? string
      |   +--rw admin-status? pcep-admin-status
      |   +--ro index? uint32
      |   +--ro oper-status? pcep-oper-status
      +--rw domain
      |   +--rw domain* [domain-type domain]
      |      ...
      +--rw capability
      |
      |   ...
      +--rw pce-info
      |   +--rw scope
      |      |
      |      +--rw neigh-domains
      |      |
      |      +--rw path-key {path-key}?
      |      |
      |      ...
      +--ro lsp-db {stateful}?
      |   +--ro db-ver? uint64
      |      (stateful-sync-opt)?
      |   +--ro association-list*
      |      [id source global-source extended-id]
      |      ...
      |   +--ro lsp* [plsp-id pcc-id lsp-id]
      |      ...
      +--ro path-keys {path-key}?
      |   +--ro path-keys* [path-key]
      |   ...
      +--rw peers
+++rw peer* [addr]
    +++ro sessions
      +++ro session* [initiator]
      ...

rpcs:
    +++x trigger-resync {stateful,stateful-sync-opt}?
        +++w input
        +++w pcc?  -> /pcep/entity/peers/peer/addr

notifications:
    +++n pcep-session-up
        ... 
    +++n pcep-session-down
        ... 
    +++n pcep-session-local-overload
        ... 
    +++n pcep-session-local-overload-clear
        ... 
    +++n pcep-session-peer-overload
        ... 
    +++n pcep-session-peer-overload-clear
        ...

5.2. The Entity

The PCEP yang module may contain status information for the local PCEP entity.

The entity has an IP address (using ietf-inet-types [RFC6991]) and a "role" leaf (the local entity PCEP role) as mandatory.

Note that, the PCEP MIB module [RFC7420] uses an entity list and a system generated entity index as a primary index to the read only entity table. If the device implements the PCEP MIB, the "index" leaf MUST contain the value of the corresponding pcePcepEntityIndex and only one entity is assumed.

The various information related to this entity such as its domain, capabilities etc. Further incase when the entity is PCE it could also have path-key and the LSP-DB information.

module: ietf-pcep
    +++rw pcep!
        +++rw entity
            +++rw addr    inet:ip-address
            +++rw enabled?    boolean

+++rw role                           pcep-role
+++rw description?                   string
+++rw speaker-entity-id?             string {sync-opt}?
+++rw admin-status?                  boolean
+++ro index?                         uint32
+++ro oper-status?                   pcep-oper-status
+++rw domain
   +++rw domain* [domain-type domain]
   |     +++rw domain-type  domain-type
   |     +++rw domain  domain
+++rw capability
   +++rw capability?                   bits
   +++rw pce-initiated?                boolean {pce-initiated}?
   |     +++rw include-db-ver?          boolean
   |     |       {stateful, sync-opt}?
   |     +++rw trigger-resync?           boolean
   |     |       {stateful, sync-opt}?
   |     +++rw trigger-initial-sync?     boolean
   |     |       {stateful, sync-opt}?
   |     +++rw incremental-sync?         boolean
   |     |       {stateful, sync-opt}?
   |     +++rw sr [sr]?
   |     |     +++rw enabled?               boolean
   |     |     +++rw msd-limit?              boolean
   |     |     +++rw nai?                    boolean
   |     +--rw msd?                       uint8 {sr}?
+++rw pce-info
   +++rw scope
   |     +++rw path-scope?               bits
   |     +++rw intra-area-pref?          uint8
   |     +++rw inter-area-pref?          uint8
   |     +++rw inter-as-pref?            uint8
   |     +++rw inter-layer-pref?         uint8
   +++rw neigh-domains
   |     +++rw domain* [domain-type domain]
   |     |     +++rw domain-type  domain-type
   |     |     +++rw domain  domain
   +++rw path-key [path-key]?
   |     +++rw enabled?                  boolean
   |     +++rw discard-timer?            uint32
   |     +++rw reuse-time?               uint32
   |     +++rw pce-id?                   inet:ip-address
   |     +++rw connect-timer?            uint16
   |     +++rw connect-max-retry?        uint32
   |     +++rw init-backoff-timer?       uint32
   |     +++rw max-backoff-timer?        uint32
   |     +++rw open-wait-timer?          uint16
   |     +++rw keep-wait-timer?          uint16

Internet-Draft                  PCE-YANG                       July 2019

+--rw keep-alive-timer?       uint8
+--rw dead-timer?             uint8
+--rw allow-negotiation?      boolean
+--rw max-keep-alive-timer?   uint8
+--rw max-dead-timer?         uint8
+--rw min-keep-alive-timer?   uint8
+--rw min-dead-timer?         uint8
+--rw sync-timer?             uint16 {svec}?
+--rw request-timer?          uint16
+--rw max-sessions?           uint32
+--rw max-unknown-reqs?       uint32
+--rw max-unknown-msgs?       uint32
+--rw pcep-notification-max-rate uint32
+--rw stateful-parameter (stateful)?
  +--rw state-timeout?         uint32
  +--rw redelegation-timeout?  uint32
  +--rw rpt-non-pcep-lsp?      boolean
+--rw of-list (objective-function)?
  +--rw objective-function* [of]
    +--rw of identityref
+--ro lsp-db (stateful)?
  +--ro db-ver?                uint64 {sync-opt}?
  +--ro association-list*     (association)?
    +--ro type identityref
    +--ro id                  uint16
    +--ro source inet:ip-address
    +--ro global-source uint32
    +--ro extended-id string
    +--ro lsp* [plsp-id pcc-id lsp-id]
      +--ro plsp-id -> /pcep/entity/lsp-db/lsp/plsp-id
      +--ro pcc-id   -> /pcep/entity/lsp-db/lsp/pcc-id
      +--ro lsp-id   -> /pcep/entity/lsp-db/lsp/lsp-id
    +--ro lsp* [plsp-id pcc-id lsp-id]
      +--ro plsp-id uint32
      +--ro pcc-id inet:ip-address
      +--ro source?
        | -> /te:te/lsps-state/lsp/source
      +--ro destination?
        | -> /te:te/lsps-state/lsp/destination
      +--ro tunnel-id?
        | -> /te:te/lsps-state/lsp/tunnel-id
      +--ro lsp-id
        | -> /te:te/lsps-state/lsp/lsp-id
      +--ro extended-tunnel-id?
        | -> /te:te/lsps-state/lsp/extended-tunnel-id
      +--ro admin-state? boolean
5.3. The Peer Lists

The peer list contains peer(s) that the local PCEP entity knows about. A PCEP speaker is identified by its IP address. If there is a PCEP speaker in the network that uses multiple IP addresses then it looks like multiple distinct peers to the other PCEP speakers in the network.

Since PCEP sessions can be ephemeral, the peer list tracks a peer even when no PCEP session currently exists to that peer. The statistics contained are an aggregate of the statistics for all successive sessions to that peer.

To limit the quantity of information that is stored, an implementation MAY choose to discard this information if and only if no PCEP session exists to the corresponding peer.

The data model for PCEP peer presented in this document uses a flat list of peers. Each peer in the list is identified by its IP address (addr-type, addr).

There is a list for static peer configuration and operational state of all peers (i.e., static as well as discovered) (/pcep/entity/peers). The list is used to enable remote PCE configuration at PCC (or PCE) and has the operational state of these peers as well as the remote PCE peer which were discovered and PCC peers that have initiated session.
++--rw entity
...
++--rw peers
++--rw peer* [addr]
  ++--rw addr           inet:ip-address
  ++--rw role           pcep-role
  ++--rw description?   string
++--rw domain
  |++--rw domain* [domain-type domain]
  |  ++--rw domain-type  domain-type
  |  ++--rw domain       domain
++--rw capability
  ++--rw capability?    bits
  ++--rw pce-initiated? boolean
  |    {pce-initiated}?
  ++--rw include-db-ver? boolean
  |    {stateful,sync-opt}?
  ++--rw trigger-resync? boolean
  |    {stateful,sync-opt}?
  ++--rw trigger-initial-sync? boolean
  |    {stateful,sync-opt}?
  ++--rw incremental-sync? boolean
  |    {stateful,sync-opt}?
  ++--rw sr [sr]?
  ++--rw msd?            uint8 (sr)?
  ++--rw msd-limit?      boolean
  ++--rw nai?            boolean
++--rw pce-info
  ++--rw scope
  |  ++--rw path-scope?   bits
  |  ++--rw intra-area-pref? uint8
  |  ++--rw inter-area-pref? uint8
  |  ++--rw inter-as-pref? uint8
  |  ++--rw inter-layer-pref? uint8
  ++--rw neigh-domains
  |  ++--rw domain* [domain-type domain]
  |  |  ++--rw domain-type  domain-type
  |  |  ++--rw domain       domain
++--rw delegation-pref? uint8 {stateful}?
++--rw auth
  ++--rw (auth-type-selection)?
  |++--:(auth-key-chain)
  |  |++--rw key-chain?
  |  |  | key-chain:key-chain-ref
  |++--:(auth-key)
  |  |++--rw crypto-algorithm identityref
  |  |++--rw (key-string-style)?
5.4. The Session Lists

The session list contains PCEP session that the PCEP entity (PCE or PCC) is currently participating in. The statistics in session are semantically different from those in peer since the former applies to the current session only, whereas the latter is the aggregate for all sessions that have existed to that peer.

Although [RFC5440] forbids more than one active PCEP session between a given pair of PCEP entities at any given time, there is a window during session establishment where two sessions may exist for a given pair, one representing a session initiated by the local PCEP entity and the other representing a session initiated by the peer. If either of these sessions reaches active state first, then the other is discarded.

The data model for PCEP session presented in this document uses a flat list of sessions. Each session in the list is identified by its initiator. This index allows two sessions to exist transiently for a given peer, as discussed above.
module: ietf-pcep
   +--rw pcep!
   +--rw entity
      ...
   +--rw peers
      +--rw peer* [addr]
      ...
   +--ro sessions
      +--ro session* [initiator]
         +--ro initiator pcep-initiator
         +--ro role?
         |   --> /pcep/entity/role
         +--ro state-last-change? yang:timestamp
         +--ro state? pcep-sess-state
         +--ro session-creation? yang:timestamp
         +--ro connect-retry? yang:counter32
         +--ro local-id? uint8
         +--ro remote-id? uint8
         +--ro keepalive-timer? uint8
         +--ro peer-keepalive-timer? uint8
         +--ro dead-timer? uint8
         +--ro peer-dead-timer? uint8
         +--ro ka-hold-time-rem? uint8
         +--ro overloaded? boolean
         +--ro overloaded-time? uint32
         +--ro peer-overloaded? boolean
         +--ro peer-overload-time? uint32
         +--ro lspdb-sync? sync-state
         |   {stateful}?
         +--ro recv-db-ver? uint64
         |   {stateful, sync-opt}?
         +--ro of-list {objective-function}?
         |   +--ro objective-function* [of]
         |   |   +--ro of identityref
         +--ro pst-list
         |   +--ro path-setup-type* [pst]
         |   |   +--ro pst identityref
         +--ro assoc-type-list {association}?
         |   +--ro assoc-type* [at]
         |   |   +--ro at identityref
         +--ro speaker-entity-id? string {sync-opt}?
         +--ro discontinuity-time? yang:timestamp
5.5. Notifications

This YANG model defines a list of notifications to inform client of important events detected during the protocol operation. The notifications defined cover the PCEP MIB notifications.

notifications:

---n pcep-session-up
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro session-initiator? --> /pcep/entity/peers/peer/sessions/session/initiator
  +--ro state-last-change? yang:timestamp
  +--ro state? pcep-sess-state

---n pcep-session-down
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro session-initiator? pcep-initiator
  +--ro state-last-change? yang:timestamp
  +--ro state? pcep-sess-state

---n pcep-session-local-overload
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro session-initiator? --> /pcep/entity/peers/peer/sessions/session/initiator
  +--ro overloaded? boolean
  +--ro overload-time? uint32

---n pcep-session-local-overload-clear
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro overloaded? boolean

---n pcep-session-peer-overload
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro session-initiator? --> /pcep/entity/peers/peer/sessions/session/initiator
  +--ro peer-overloaded? boolean
  +--ro peer-overload-time? uint32

---n pcep-session-peer-overload-clear
  +--ro peer-addr? --> /pcep/entity/peers/peer/addr
  +--ro peer-overloaded? boolean

5.6. RPC

This YANG model defines a RPC to trigger state resynchronization to a particular PCEP peer.

rpcs:

---x trigger-resync {stateful, sync-opt}?
  +--w input
    +--w pcc? --> /pcep/entity/peers/peer/addr
5.7. The Full PCEP Data Model

The module, "ietf-pcep", defines the basic components of a PCE speaker. The tree depth in the tree is set to 10.

module: ietf-pcep
   +--rw pcep
       +--rw entity
           +--rw addr inet:ip-address
           +--rw enabled? boolean
           +--rw role pcep-role
           +--rw description? string
           +--rw speaker-entity-id? string {sync-opt}?
           +--rw admin-status? boolean
           +--ro index? uint32
           +--ro oper-status? pcep-oper-status
           +--rw domain
               +--rw domain* [domain-type domain]
                 +--rw domain-type domain-type
                 +--rw domain domain
           +--rw capability
               +--rw capability? bits
               +--rw pce-initiated? boolean {pce-initiated}?
               +--rw include-db-ver? boolean
               |   (stateful, sync-opt)?
               +--rw trigger-resync? boolean
               |   (stateful, sync-opt)?
               +--rw trigger-initial-sync? boolean
               |   (stateful, sync-opt)?
               +--rw incremental-sync? boolean
               |   (stateful, sync-opt)?
               +--rw sr (sr)?
               |   +--rw enabled? boolean
               |   +--rw msd-limit? boolean
               |   +--rw nai? boolean
           +--rw msd? uint8 {sr}?
       +--rw pce-info
           +--rw scope
               +--rw path-scope? bits
               +--rw intra-area-pref? uint8
               +--rw inter-area-pref? uint8
               +--rw inter-as-pref? uint8
               +--rw inter-layer-pref? uint8
           +--rw neigh-domains
               +--rw domain* [domain-type domain]
                 +--rw domain-type domain-type
                 +--rw domain domain
           +--rw path-key {path-key}?
Internet-Draft                  PCE-YANG                       July 2019

|     +--rw enabled?         boolean
|     +--rw discard-timer?   uint32
|     +--rw reuse-time?      uint32
|     +--rw pce-id?          inet:ip-address
|     +--rw connect-timer?                uint16
|     +--rw connect-max-retry?            uint32
|     +--rw init-backoff-timer?           uint16
|     +--rw max-backoff-timer?            uint32
|     +--rw open-wait-timer?             uint16
|     +--rw keep-wait-timer?             uint16
|     +--rw keep-alive-timer?            uint8
|     +--rw dead-timer?                uint8
|     +--rw allow-negotiation?          boolean
|     +--rw max-keep-alive-timer?         uint8
|     +--rw max-dead-timer?              uint8
|     +--rw min-keep-alive-timer?         uint8
|     +--rw min-dead-timer?              uint8
|     +--rw sync-timer?                 uint16 {svec}? 
|     +--rw request-timer?              uint16
|     +--rw max-sessions?               uint32
|     +--rw max-unknown-reqs?            uint32
|     +--rw max-unknown-msgs?            uint32
|     +--rw pcep-notification-max-rate    uint32
+--rw stateful-parameter {stateful}?  
|     +--rw state-timeout?             uint32
|     +--rw redelegation-timeout?       uint32
|     +--rw rpt-non-pcep-lsp?           boolean
+--rw of-list {objective-function}?  
|     +--rw objective-function* [of]   
|     |     +--rw of identityref
|     +--ro lsp-db {stateful}?          
|     |     +--ro db-ver?             uint64 {sync-opt}?  
|     |     +--ro association-list*   
|     |     |       [type id source global-source extended-id] 
|     |     |       (association)?
|     |     |       +--ro type identityref
|     |     |       +--ro id uint16
|     |     |       +--ro source inet:ip-address
|     |     |       +--ro global-source uint32
|     |     |       +--ro extended-id string
|     |     |       +--ro lsp* [plsp-id pcc-id lsp-id]
|     |     |       |       +--ro plsp-id
|     |     |       |       |       -> /pcep/entity/lsp-db/lsp/plsp-id
|     |     |       |       +--ro pcc-id -> /pcep/entity/lsp-db/lsp/pcc-id
|     |     |       |       +--ro lsp-id -> /pcep/entity/lsp-db/lsp/lsp-id
|     |     |       +--ro lsp* [plsp-id pcc-id lsp-id]
|     |     |       +--ro plsp-id uint32
|     |     |       +--ro pcc-id inet:ip-address
# Internet-Draft

## PCE-YANG

July 2019

```yang
++-ro source?
    |   -> /te:te/lsps-state/lsp/source
++-ro destination?
    |   -> /te:te/lsps-state/lsp/destination
++-ro tunnel-id?
    |   -> /te:te/lsps-state/lsp/tunnel-id
++-ro lsp-id
    |   -> /te:te/lsps-state/lsp/lsp-id
++-ro extended-tunnel-id?
    |   -> /te:te/lsps-state/lsp/extended-tunnel-id
++-ro admin-state? boolean
++-ro operational-state? operational-state
++-ro delegated
++-ro enabled? boolean
++-ro peer?   -> /pcep/entity/peers/peer/addr
++-ro srp-id? uint32
++-ro initiation [pce-initiated]?
++-ro enabled? boolean
++-ro peer?   -> /pcep/entity/peers/peer/addr
++-ro symbolic-path-name? string
++-ro last-error? identityref
++-ro pst? identityref
++-ro association-list*
    | [type id source global-source extended-id] (association)?
++-ro type leafref
++-ro id
    |   -> /pcep/entity/lsp-db/association-list/id
++-ro source leafref
++-ro global-source leafref
++-ro extended-id leafref
++-ro path-keys [path-key]?
++-ro path-keys* [path-key]
++-ro path-key uint16
++-ro cps
++-ro explicit-route-objects* [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)
```

Dhody, et al. Expires January 2, 2020
| +--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
| ...\n\n+--ro pcc-original?  \n|   -> /pcep/entity/peers/peer/addr
+--ro req-id?         uint32
+--ro retrieved?      boolean
+--ro pcc-retrieved?  \n|   -> /pcep/entity/peers/peer/addr
+--ro creation-time?  yang:timestamp
+--ro discard-time?   uint32
+--ro reuse-time?     uint32
+--rw peers
+--rw peer* [addr]
| +--rw addr             inet:ip-address
| +--rw role             pcep-role
| +--rw description?     string
| +--rw domain
| | +--rw domain* [domain-type domain]
| |    +--rw domain-type    domain-type
| |    +--rw domain         domain
| +--rw capability
| | +--rw capability?      bits
| |    +--rw pce-initiated? boolean
| |    | {pce-initiated}?
| |    +--rw include-db-ver? boolean
| |    | {stateful, sync-opt}?
| |    +--rw trigger-resync? boolean
| |    | {stateful, sync-opt}?
| |    +--rw trigger-initial-sync? boolean
| |    | {stateful, sync-opt}?
| |    +--rw incremental-sync? boolean
| |    | {stateful, sync-opt}?
| |    +--rw sr {sr}?
| |    | +--rw enabled? boolean
| |    | +--rw msd-limit? boolean
| |    | +--rw nai? boolean
| |    +--rw msd? uint8 {sr}?
++-rw pce-info
  +++-rw scope
    |  +++-rw path-scope?      bits
    |  +++-rw intra-area-pref? uint8
    |  +++-rw inter-area-pref? uint8
    |  +++-rw inter-as-pref?   uint8
    |  +++-rw inter-layer-pref? uint8
  +++-rw neigh-domains
    |  +++-rw domain* [domain-type domain]
    |     +++-rw domain-type domain-type
    |     +++-rw domain domain
  +++-rw delegation-pref? uint8 {stateful}?
++-rw auth
  +++-rw (auth-type-selection)?
    |  +++:(auth-key-chain)
      |    |    +++-rw key-chain?
      |    |      key-chain:key-chain-ref
    |  +++:(auth-key)
    |  +++-rw crypto-algorithm
      |      identityref
    |  +++-rw (key-string-style)?
      |    +++:(keystring)
      |    |    +++-rw keystring? string
    |    +++-(hexadecimal)
      |      |    [{key-chain:hex-key-string}?
      |      |        +++-rw hexadecimal-string? yang:hex-string
    |    +++:(auth-tls) {tls}?
      |    +++-(server)
      |      +++-rw tls-server
      |    +++-(client)
      |      +++-rw tls-client
    |  +++-ro (role)?
      |    +++:(server)
      |      +++-rw tls-server
      |    +++-(client)
      |      +++-rw tls-client
    |  +++-ro discontinuity-time? yang:timestamp
  +++-ro initiate-session? boolean
  +++-ro session-exists? boolean
  +++-ro session-up-time? yang:timestamp
  +++-ro session-fail-time? yang:timestamp
  +++-ro session-fail-up-time? yang:timestamp
  +++-ro sessions
    |  +++-rw session* [initiator]
    |    |    +++-rw initiator pcep-initiator
    |    |    +++-rw role?
    |    |      -> /pcep/entity/role
    |    |    +++-ro state-last-change? yang:timestamp
    |    |    +++-ro state? pcep-sess-state
++-ro session-creation?       yang:timestamp
++-ro connect-retry?          yang:counter32
++-ro local-id?               uint8
++-ro remote-id?              uint8
++-ro keepalive-timer?        uint8
++-ro peer-keepalive-timer?   uint8
++-ro dead-timer?             uint8
++-ro peer-dead-timer?        uint8
++-ro ka-hold-time-rem?       uint8
++-ro overloaded?             boolean
++-ro overload-time?          uint32
++-ro peer-overloaded?        boolean
++-ro peer-overload-time?     uint32
++-ro lspdb-sync?             sync-state
|       {stateful}?            
|       ++-ro recv-db-ver?     uint64
|       {stateful, sync-opt}?  
++-ro of-list {objective-function}?  
|       +++-ro objective-function* [of] 
|       |       ++-ro of           identityref
++-ro pst-list                 
|       ++-ro path-setup-type* [pst] 
|       |       ++-ro pst           identityref
++-ro assoc-type-list {association}?  
|       +++-ro assoc-type* [at] 
|       |       +++-ro at           identityref
++-ro speaker-entity-id?      string
|       {sync-opt}?            
++-ro discontinuity-time?     yang:timestamp

rpcs:
    +++-x trigger-resync {stateful, sync-opt}?
    +++-w input
         +++-w pcc?   -> /pcep/entity/peers/peer/addr

notifications:
    +++-n pcep-session-up
        |       +++-ro peer-addr?     -> /pcep/entity/peers/peer/addr
        |       +++-ro session-initiator?
        |           |       -> /pcep/entity/peers/peer/sessions/session/initiator
        |       +++-ro state-last-change? yang:timestamp
        |       +++-ro state?          pcep-sess-state
    +++-n pcep-session-down
        |       +++-ro peer-addr?     -> /pcep/entity/peers/peer/addr
        |       +++-ro session-initiator? pcep-initiator
        |       +++-ro state-last-change? yang:timestamp
        |       +++-ro state?          pcep-sess-state
    +++-n pcep-session-local-overload
6. The Design of PCEP Statistics Data Model

The module, "ietf-pcep-stats", augments the ietf-pcep module to include statistics at the PCEP peer and session level.

module: ietf-pcep-stats
augment /pcep:pcep/pcep:entity/pcep:peers/pcep:peer:
  +--ro num-sess-setup-ok?     yang:counter32
  +--ro num-sess-setup-fail?   yang:counter32
  +--ro pcep-stats
    +--ro avg-rsp-time?               uint32
    +--ro lwm-rsp-time?               uint32
    +--ro hwm-rsp-time?               uint32
    +--ro num-pcreq-sent?             yang:counter32
    +--ro num-pcreq-rcvd?             yang:counter32
    +--ro num-pcrep-sent?             yang:counter32
    +--ro num-pcrep-rcvd?             yang:counter32
    +--ro num-pcerr-sent?             yang:counter32
    +--ro num-pcerr-rcvd?             yang:counter32
    +--ro num-pcntf-sent?             yang:counter32
    +--ro num-pcntf-rcvd?             yang:counter32
    +--ro num-keepalive-sent?         yang:counter32
    +--ro num-keepalive-rcvd?         yang:counter32
    +--ro num-unknown-rcvd?           yang:counter32
    +--ro num-req-sent?               yang:counter32
    +--ro num-req-sent-pend-rep?      yang:counter32
++--ro num-req-sent-ero-rcvd? yang:counter32
++--ro num-req-sent-nopath-rcvd? yang:counter32
++--ro num-req-sent-cancel-rcvd? yang:counter32
++--ro num-req-sent-error-rcvd? yang:counter32
++--ro num-req-sent-timeout? yang:counter32
++--ro num-req-sent-cancel-sent? yang:counter32
++--ro num-req-rcvd? yang:counter32
++--ro num-req-rcvd-pend-rep? yang:counter32
++--ro num-req-rcvd-ero-sent? yang:counter32
++--ro num-req-rcvd-nopath-sent? yang:counter32
++--ro num-req-rcvd-cancel-sent? yang:counter32
++--ro num-req-rcvd-error-sent? yang:counter32
++--ro num-req-rcvd-unknown? yang:counter32
++--ro num-req-rcvd-unknown? yang:counter32
++--ro svec {pcep:svec}?
  +--ro num-svec-sent? yang:counter32
  +--ro num-svec-req-sent? yang:counter32
  +--ro num-svec-rcvd? yang:counter32
  +--ro num-svec-req-rcvd? yang:counter32
++--ro stateful {pcep:stateful}?
  +--ro num-pcrpt-sent? yang:counter32
  +--ro num-pcrpt-rcvd? yang:counter32
  +--ro num-pcupd-sent? yang:counter32
  +--ro num-pcupd-rcvd? yang:counter32
  +--ro num-rpt-sent? yang:counter32
  +--ro num-rpt-rcvd? yang:counter32
  +--ro num-rpt-rcvd-error-sent? yang:counter32
  +--ro num-upd-sent? yang:counter32
  +--ro num-upd-rcvd? yang:counter32
  +--ro num-upd-rcvd-unknown? yang:counter32
  +--ro num-upd-rcvd-undelegated? yang:counter32
  +--ro num-upd-rcvd-error-sent? yang:counter32
  +--ro initiation {pcep:pce-initiated}?
    +--ro num-pcinitiate-sent? yang:counter32
    +--ro num-pcinitiate-rcvd? yang:counter32
    +--ro num-initiate-sent? yang:counter32
    +--ro num-initiate-rcvd? yang:counter32
    +--ro num-initiate-rcvd-error-sent? yang:counter32
  +--ro path-key {pcep:path-key}?
    +--ro num-unknown-path-key? yang:counter32
    +--ro num-exp-path-key? yang:counter32
    +--ro num-dup-path-key? yang:counter32
    +--ro num-path-key-no-attempt? yang:counter32
    +--ro num-req-sent-closed? yang:counter32
    +--ro num-req-rcvd-closed? yang:counter32
---ro pcep-stats
  +--ro avg-rsp-time?                  uint32
  +--ro lwm-rsp-time?                  uint32
  +--ro hwm-rsp-time?                  uint32
  +--ro num-pcreq-sent?               yang:counter32
  +--ro num-pcreq-rcvd?               yang:counter32
  +--ro num-pcrep-sent?               yang:counter32
  +--ro num-pcrep-rcvd?               yang:counter32
  +--ro num-pcerr-sent?               yang:counter32
  +--ro num-pcerr-rcvd?               yang:counter32
  +--ro num-pcntf-sent?               yang:counter32
  +--ro num-pcntf-rcvd?               yang:counter32
  +--ro num-keepalive-sent?           yang:counter32
  +--ro num-keepalive-rcvd?           yang:counter32
  +--ro num-unknown-rcvd?             yang:counter32
  +--ro num-req-sent?                 yang:counter32
  +--ro num-req-sent-pend-rep?        yang:counter32
  +--ro num-req-sent-ero-rcvd?        yang:counter32
  +--ro num-req-sent-nopath-rcvd?     yang:counter32
  +--ro num-req-sent-cancel-rcvd?     yang:counter32
  +--ro num-req-sent-error-rcvd?      yang:counter32
  +--ro num-req-sent-timeout?         yang:counter32
  +--ro num-req-sent-cancel-sent?     yang:counter32
  +--ro num-req-rcvd?                 yang:counter32
  +--ro num-req-rcvd-pend-rep?        yang:counter32
  +--ro num-req-rcvd-ero-sent?        yang:counter32
  +--ro num-req-rcvd-nopath-sent?     yang:counter32
  +--ro num-req-rcvd-cancel-sent?     yang:counter32
  +--ro num-req-rcvd-cancel-rcvd?     yang:counter32
  +--ro num-req-rcvd-error-sent?      yang:counter32
  +--ro num-req-rcvd-unknown?         yang:counter32
  +--ro num-svec {pcep:svec}?         yang:counter32
    | +--ro num-svec-sent?               yang:counter32
    | +--ro num-svec-req-sent?           yang:counter32
    | +--ro num-svec-rcvd?               yang:counter32
    | +--ro num-svec-req-rcvd?           yang:counter32
  +--ro stateful {pcep:stateful}?     yang:counter32
    | +--ro num-pcrpt-sent?              yang:counter32
    | +--ro num-pcrpt-rcvd?              yang:counter32
    | +--ro num-pcupd-sent?              yang:counter32
    | +--ro num-pcupd-rcvd?              yang:counter32
    | +--ro num-rpt-sent?                yang:counter32
    | +--ro num-rpt-rcvd?                yang:counter32
    | +--ro num-upd-sent?                yang:counter32
    | +--ro num-upd-rcvd?                yang:counter32
7. Advanced PCE Features

This document contains a specification of the base PCEP YANG module, "ietf-pcep" which provides the basic PCEP [RFC5440] data model.

This document further handles advanced PCE features like -

- Capability and Scope
- Domain information (local/neighbour)
- Path-Key
- Objective Function (OF)
- Global Concurrent Optimization (GCO)
- P2MP
- GMPLS
- Inter-Layer
- Stateful PCE
- Segment Routing (SR)
- Authentication including PCEPS (TLS)
7.1. Stateful PCE’s LSP-DB

In the operational state of PCEP which supports stateful PCE mode, the list of LSP state are maintained in LSP-DB. The key is the PLSP-ID and the PCC IP address.

The PCEP data model contains the operational state of LSPs (/pcep/entity/lsp-db/lsp/) with PCEP specific attributes. The generic TE attributes of the LSP are defined in [I-D.ietf-teas-yang-te]. A reference to LSP state in TE model is maintained.

8. Open Issues and Next Step

This section is added so that open issues can be tracked. This section would be removed when the document is ready for publication.

8.1. The PCE-Initiated LSP

The TE Model at [I-D.ietf-teas-yang-te] should support creating of tunnels at the controller (PCE) and marking them as PCE-Initiated. The LSP-DB in the PCEP Yang (/pcep/entity/lsp-db/lsp/initiation) also marks the LSPs which are PCE-initiated.

9. Other Considerations

9.1. PCEP over TLS (PCEPS)

[RFC8253] describe the use of TLS in PCEP. The peer acting as the PCEP client MUST act as the TLS client. The TLS client actively opens the TLS connection and the TLS server passively listens for the incoming TLS connections. The well-known TCP port number 4189 is used by PCEP servers to listen for TCP connections established by PCEP over TLS clients. The TLS client MUST send the TLS ClientHello message to begin the TLS handshake. The TLS server MUST send a CertificateRequest in order to request a certificate from the TLS client. Once the TLS handshake has finished, the client and the server MAY begin to exchange PCEP messages. Client and server identity verification is done before the PCEP open message is sent. This means that the identity verification is completed before the PCEP session is started.

10. PCEP YANG Modules
10.1. ietf-pcep module

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

<CODE BEGINS> file "ietf-pcep@2019-07-01.yang"
module ietf-pcep {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pcep";
  prefix pcep;
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-te {
    prefix te;
    reference
      "I-D.ietf-teas-yang-te: A YANG Data Model for Traffic Engineering Tunnels and Interfaces";
  }
  import ietf-te-types {
    prefix te-types;
    reference
      "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
  }
  import ietf-key-chain {
    prefix key-chain;
    reference
      "RFC 8177: YANG Data Model for Key Chains";
  }
  import ietf-netconf-acm {
    prefix nacm;
    reference
      "RFC 8341: Network Configuration Protocol (NETCONF) Access Control Model";
  }
}

import ietf-tls-server {
  prefix tlss;
  reference
    "I-D.ietf-netconf-tls-client-server: YANG Groupings for TLS
     Clients and TLS Servers";
}
import ietf-tls-client {
  prefix tlsc;
  reference
    "I-D.ietf-netconf-tls-client-server: YANG Groupings for TLS
     Clients and TLS Servers";
}
import ietf-ospf {
  prefix ospf;
  reference
    "I-D.ietf-ospf-yang: YANG Data Model for OSPF Protocol";
}
import ietf-isis {
  prefix isis;
  reference
    "I-D.ietf-isis-yang-isis-cfg: YANG Data Model for IS-IS
     Protocol";
}

organization
  "IETF PCE (Path Computation Element) Working Group";
contact
  "WG Web:  <https://tools.ietf.org/wg/pce/>
  WG List:  <mailto:pce@ietf.org>
  Editor:   Dhruv Dhody
            <mailto:dhruv.ietf@gmail.com>"

description
  "The YANG module defines a generic configuration and
   operational model for PCEP.

  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

  This version of this YANG module is part of RFC XXXX; see the
  RFC itself for full legal notices.";
typedef pcep-role {
  type enumeration {
    enum unknown {
      value 0;
      description
        "An unknown role";
    }
    enum pcc {
      value 1;
      description
        "The role of a Path Computation Client";
    }
    enum pce {
      value 2;
      description
        "The role of Path Computation Element";
    }
    enum pcc-and-pce {
      value 3;
      description
        "The role of both Path Computation Client and Path Computation Element";
    }
  }
} description
"The role of a PCEP speaker.
Takes one of the following values
- unknown(0): the role is not known.
- pcc(1): the role is of a Path Computation Client (PCC).
- pce(2): the role is of a Path Computation Server (PCE).
- pccAndPce(3): the role is of both a PCC and a PCE.";
typedef pcep-oper-status {
  type enumeration {
    enum oper-status-up {
      value 1;
      description
      "The PCEP entity is active";
    }
    enum oper-status-down {
      value 2;
      description
      "The PCEP entity is inactive";
    }
    enum oper-status-going-up {
      value 3;
      description
      "The PCEP entity is activating";
    }
    enum oper-status-going-down {
      value 4;
      description
      "The PCEP entity is deactivating";
    }
    enum oper-status-failed {
      value 5;
      description
      "The PCEP entity has failed and will recover when possible.";
    }
    enum oper-status-failed-perm {
      value 6;
      description
      "The PCEP entity has failed and will not recover without operator intervention";
    }
  }
}

description
"The operational status of the PCEP entity. Takes one of the following values
- oper-status-up(1): Active
- oper-status-down(2): Inactive
- oper-status-going-up(3): Activating
- oper-status-going-down(4): Deactivating
- oper-status-failed(5): Failed
- oper-status-failed-perm(6): Failed Permanently"
reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
typedef pcep-initiator {
    type enumeration {
        enum local {
            value 1;
            description
            "The local PCEP entity initiated the session";
        }
        enum remote {
            value 2;
            description
            "The remote PCEP peer initiated the session";
        }
    }
    description
    "The initiator of the session, that is, whether the TCP
    connection was initiated by the local PCEP entity or
    the remote peer.
    Takes one of the following values
    - local(1): Initiated locally
    - remote(2): Initiated remotely";
}

typedef pcep-sess-state {
    type enumeration {
        enum tcp-pending {
            value 1;
            description
            "The tcp-pending state of PCEP session.";
        }
        enum open-wait {
            value 2;
            description
            "The open-wait state of PCEP session.";
        }
        enum keep-wait {
            value 3;
            description
            "The keep-wait state of PCEP session.";
        }
        enum session-up {
            value 4;
            description
            "The session-up state of PCEP session.";
        }
    }
    description
    "The current state of the session.
    The set of possible states excludes the idle state
since entries do not exist in the idle state.
Takes one of the following values
- tcp-pending(1): PCEP TCP Pending state
- open-wait(2): PCEP Open Wait state
- keep-wait(3): PCEP Keep Wait state
- session-up(4): PCEP Session Up state"
reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
)

typedef domain-type {
  type enumeration {
    enum ospf-area {
      value 1;
      description "The OSPF area.";
    }
    enum isis-area {
      value 2;
      description "The IS-IS area.";
    }
    enum as {
      value 3;
      description "The Autonomous System (AS).";
    }
  }
  description "The PCE Domain Type";
}

typedef domain-ospf-area {
  type ospf:area-id-type;
  description "OSPF Area ID.";
  reference "I-D.ietf-ospf-yang: YANG Data Model for OSPF Protocol";
}

typedef domain-isis-area {
  type isis:area-address;
  description "IS-IS Area ID.";
  reference "I-D.ietf-isis-yang-isis-cfg: YANG Data Model for IS-IS Protocol";
}
typedef domain-as {
  type inet:as-number;
  description "Autonomous System number.";
}

typedef domain {
  type union {
    type domain-ospf-area;
    type domain-isis-area;
    type domain-as;
  }
  description "The Domain Information";
}

typedef operational-state {
  type enumeration {
    enum down {
      value 0;
      description "not active.";
    }
    enum up {
      value 1;
      description "signalled.";
    }
    enum active {
      value 2;
      description "up and carrying traffic.";
    }
    enum going-down {
      value 3;
      description "LSP is being torn down, resources are being released.";
    }
    enum going-up {
      value 4;
      description "LSP is being signalled.";
    }
  }
  description
}
typedef sync-state {
    type enumeration {
        enum pending {
            value 0;
            description "The state synchronization has not started.";
        }
        enum ongoing {
            value 1;
            description "The state synchronization is ongoing.";
        }
        enum finished {
            value 2;
            description "The state synchronization is finished.";
        }
    }
    description "The LSP-DB state synchronization operational status.";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

/*
 * Features
 */

feature svec {
    description "Support synchronized path computation.";
    reference "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}

feature gmpls {
description
  "Support GMPLS.";
reference
  "I-D.ietf-pce-gmpls-pcep-extensions: PCEP extensions for GMPLS";
}

feature objective-function {
  description
    "Support OF as per RFC 5541.";
  reference
    "RFC 5541: Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)";
}

feature global-concurrent {
  description
    "Support GCO as per RFC 5557.";
  reference
}

feature path-key {
  description
    "Support path-key as per RFC 5520.";
  reference
    "RFC 5520: Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism";
}

feature p2mp {
  description
    "Support P2MP as per RFC 8306.";
  reference
    "RFC 8306: Extensions to the Path Computation Element Communication Protocol (PCEP) for Point-to-Multipoint Traffic Engineering Label Switched Paths";
}

feature stateful {
  description
    "Support stateful PCE as per RFC 8231.";
  reference
    "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}
feature sync-opt {
    description "Support stateful sync optimization as per RFC 8232";
    reference "RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}

feature pce-initiated {
    description "Support PCE-Initiated LSP as per RFC 8281.";
}

feature tls {
    description "Support PCEP over TLS as per RFC 8253.";
    reference "RFC 8253: PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)";
}

feature sr {
    description "Support Segment Routing for PCE.";
    reference "I-D.ietf-pce-segment-routing: PCEP Extensions for Segment Routing";
}

feature association {
    description "Support Association in PCE.";
    reference "I-D.ietf-pce-association-group: PCEP Extensions for Establishing Relationships Between Sets of LSPs";
}

/*
 * Identities
 */
identity lsp-error {
    if-feature "stateful";
    description "Base LSP error";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity no-error-lsp-error {
    base lsp-error;
    if-feature "stateful";
    description "No error, LSP is fine.";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity unknown-lsp-error {
    base lsp-error;
    if-feature "stateful";
    description "Unknown reason.";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity limit-lsp-error {
    base lsp-error;
    if-feature "stateful";
    description "Limit reached for PCE-controlled LSPs.";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity pending-lsp-error {
    base lsp-error;
    if-feature "stateful";
    description "Too many pending LSP update requests.";
    reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}
identity unacceptable-lsp-error {
  base lsp-error;
  if-feature "stateful";
  description "Unacceptable parameters.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity internal-lsp-error {
  base lsp-error;
  if-feature "stateful";
  description "Internal error.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity admin-lsp-error {
  base lsp-error;
  if-feature "stateful";
  description "LSP administratively brought down.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity preempted-lsp-error {
  base lsp-error;
  if-feature "stateful";
  description "LSP preempted.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

identity rsvp-lsp-error {
  base lsp-error;
  if-feature "stateful";
  description "RSVP signaling error.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}
grouping pce-scope {
    description
        "This grouping defines PCE path computation scope
         information which maybe relevant to PCE selection.
         This information corresponds to PCE auto-discovery
         information.";
    reference
        "RFC 5088: OSPF Protocol Extensions for Path
         Computation Element (PCE) Discovery
         RFC 5089: IS-IS Protocol Extensions for Path
         Computation Element (PCE) Discovery";
    leaf path-scope {
        type bits {
            bit intra-area-scope {
                description
                    "PCE can compute intra-area paths.";
            }
            bit inter-area-scope {
                description
                    "PCE can compute inter-area paths.";
            }
            bit inter-area-scope-default {
                description
                    "PCE can act as a default PCE for inter-area
                     path computation.";
            }
            bit inter-as-scope {
                description
                    "PCE can compute inter-AS paths.";
            }
            bit inter-as-scope-default {
                description
                    "PCE can act as a default PCE for inter-AS
                     path computation.";
            }
            bit inter-layer-scope {
                description
                    "PCE can compute inter-layer paths.";
            }
        }
        description
            "The field corresponding to the path scope bits";
    }
}
leaf intra-area-pref {
  type uint8 {
    range "0..7";
  }
  description
    "The PCE’s preference for intra-area TE LSP computation.";
}

leaf inter-area-pref {
  type uint8 {
    range "0..7";
  }
  description
    "The PCE’s preference for inter-area TE LSP computation.";
}

leaf inter-as-pref {
  type uint8 {
    range "0..7";
  }
  description
    "The PCE’s preference for inter-AS TE LSP computation.";
}

leaf inter-layer-pref {
  type uint8 {
    range "0..7";
  }
  description
    "The PCE’s preference for inter-layer TE LSP computation.";
}

//pce-scope

grouping domain {
  description
    "This grouping specifies a Domain where the PCEP speaker has topology visibility.";
  leaf domain-type {
    type domain-type;
    description
      "The domain type.";
  }
  leaf domain {
    type domain;
  }
}
description
"The domain Information.";
}
}

//domain

grouping info {

description
"This grouping specifies all information which maybe relevant to both PCC and PCE.
This information corresponds to PCE auto-discovery information.";

cariable domain {

description
"The local domain for the PCEP entity";

cariable domain {


domain {


description
"The local domain.";


cariable domain {


description
"The local domain for the PCEP entity.";

}
}
}

cariable capability {

description
"The PCEP entity capability information of local PCEP entity. This maybe relevant to PCE selection as well. This information corresponds to PCE auto-discovery information.";

reference
"RFC 5088: OSPF Protocol Extensions for Path Computation Element (PCE) Discovery
RFC 5089: IS-IS Protocol Extensions for Path Computation Element (PCE) Discovery";

leaf capability {

type bits {


description
"Path computation with GMPLS link constraints.";

}


description
"Bidirectional path computation.";


dhody, et al. expires January 2, 2020
bit diverse {
    description
    "Diverse path computation.";
}

bit load-balance {
    description
    "Load-balanced path computation.";
}

bit synchronize {
    if-feature "svec";
    description
    "Synchronized paths computation.";
}

bit objective-function {
    if-feature "objective-function";
    description
    "Support for multiple objective functions.";
}

bit add-path-constraint {
    description
    "Support for additive path constraints (max hop count, etc.).";
}

bit prioritization {
    description
    "Support for request prioritization.";
}

bit multi-request {
    description
    "Support for multiple requests per message.";
}

bit global-concurrent {
    if-feature "global-concurrent";
    description
    "Support for Global Concurrent Optimization (GCO).";
}

bit p2mp {
    if-feature "p2mp";
    description
    "Support for P2MP path computation.";
}

bit active {
    if-feature "stateful";
    description
    "Support for active stateful PCE.";
}
bit passive {
  if-feature "stateful";
  description
      "Support for passive stateful PCE.";
}

bit p2mp-active {
  if-feature "stateful";
  if-feature "p2mp";
  description
    "Support for active stateful PCE for P2MP.";
}

bit p2mp-passive {
  if-feature "stateful";
  if-feature "p2mp";
  description
    "Support for passive stateful PCE for P2MP.";
}

bit p2mp-pce-initiated {
  if-feature "stateful";
  if-feature "pce-initiated";
  if-feature "p2mp";
  description
    "Support for PCE-initiated LSP for P2MP.";
}

description
  "The bits string indicating the capabilities";
reference
  "RFC 5088: OSPF Protocol Extensions for Path Computation Element (PCE) Discovery
   RFC 5089: IS-IS Protocol Extensions for Path Computation Element (PCE) Discovery";
}

leaf pce-initiated {
  if-feature "pce-initiated";
  type boolean;
  description
    "Set to true if PCE-initiated LSP capability is enabled.";
reference
  "RFC 8281: Path Computation Element Communication Protocol (PCEP) Extensions for PCE-Initiated LSP Setup in a Stateful PCE Model";
}

leaf include-db-ver {
  if-feature "stateful";
  if-feature "sync-opt";
  type boolean;
description
"Support inclusion of LSP-DB-VERSION in LSP object";
reference
"RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}
leaf trigger-resync {
  if-feature "stateful";
  if-feature "sync-opt";
  type boolean;
  description
  "Support PCE triggered re-synchronization";
  reference
  "RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}
leaf trigger-initial-sync {
  if-feature "stateful";
  if-feature "sync-opt";
  type boolean;
  description
  "PCE triggered initial synchronization";
  reference
  "RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}
leaf incremental-sync {
  if-feature "stateful";
  if-feature "sync-opt";
  type boolean;
  description
  "Support incremental (delta) sync";
  reference
  "RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}
container sr {
  if-feature "sr";
  description
  "If segment routing is supported";
  reference
  "I-D.ietf-pce-segment-routing: PCEP Extensions for Segment Routing";
  leaf enabled {
    type boolean;
    description
    "Set to true if SR is enabled";
  }
}
leaf msd-limit {
    type boolean;
    default "false";
    description
        "True indicates no limit on MSD, the 
         leaf msd is ignored";
}
leaf nai {
    type boolean;
    default "false";
    description
        "True indicates capability to resolve NAI 
         to SID";
}
//sr
//capability
leaf msd {
    if-feature "sr";
    type uint8;
    description
        "Maximum SID Depth for SR";
    reference
        "I-D.ietf-pce-segment-routing: PCEP Extensions for Segment 
         Routing";
}
//info
grouping pce-info {
    description
        "This grouping specifies all PCE information 
         which maybe relevant to the PCE selection. 
         This information corresponds to PCE auto-discovery 
         information.";
    container scope {
        description
            "The path computation scope";
        uses pce-scope;
    }
    container neigh-domains {
        description
            "The list of neighbour PCE-Domain 
             toward which a PCE can compute 
             paths";
    }
list domain {
    key "domain-type domain";
    description
        "The neighbour domain.";
    uses domain {
        description
            "The PCE neighbour domain.";
    }
}
}

//pce-info

grouping notification-instance-hdr {
    description
        "This group describes common instance specific data
        for notifications.";
    leaf peer-addr {
        type leafref {
            path "/pcep/entity/peers/peer/addr";
        }
        description
            "Reference to peer address";
    }
}

// notification-instance-hdr

grouping notification-session-hdr {
    description
        "This group describes common session instance specific
        data for notifications.";
    leaf session-initiator {
        type leafref {
            path "/pcep/entity/peers/peer/sessions/"
                + "session/initiator";
        }
        description
            "Reference to pcep session initiator leaf";
    }
}

// notification-session-hdr

grouping of-list {
    description
        "List of OP";
}
list objective-function {
  key "of";
  description "The list of authorized OF";
  leaf of {
    type identityref {
      base te-types:objective-function-type;
      description "The OF authorized";
    }
  }
}

/*
 * Configuration data nodes
 */

container pcep {
  presence "The PCEP is enabled";
  description "Parameters for list of configured PCEP entities on the device.";
  container entity {
    description "The configured PCEP entity on the device.";
    leaf addr {
      type inet:ip-address;
      mandatory true;
      description "The local Internet address of this PCEP entity. If operating as a PCE server, the PCEP entity listens on this address. If operating as a PCC, the PCEP entity binds outgoing TCP connections to this address. It is possible for the PCEP entity to operate both as a PCC and a PCE Server, in which case it uses this address both to listen for incoming TCP connections and to bind outgoing TCP connections.";
    }
    leaf enabled {
      type boolean;
      default "true";
      description "The administrative status of this PCEP Entity.";
    }
  }
}
leaf role {
    type pcep-role;
    mandatory true;
    description
    "The role that this entity can play. Takes one of the following values.
    - unknown(0): this PCEP Entity role is not known.
    - pcc(1): this PCEP Entity is a PCC.
    - pce(2): this PCEP Entity is a PCE.
    - pcc-and-pce(3): this PCEP Entity is both a PCC and a PCE."
}

leaf description {
    type string;
    description
    "Description of the PCEP entity configured by the user";
}

leaf speaker-entity-id {
    if-feature "sync-opt";
    type string;
    description
    "The Speaker Entity Identifier";
    reference
    "RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}

leaf admin-status {
    type boolean;
    default "true";
    description
    "The administrative status of this PCEP Entity. The value true represents admin status as up. This is the desired operational status as currently set by an operator or by default in the implementation. The value of oper-status represents the current status of an attempt to reach this desired status."
}

leaf index {
    type uint32;
    config false;
    description
    "The index of the operational PCEP entity";
}
leaf oper-status {
  type pcep-oper-status;
  config false;
  description
    "The operational status of the PCEP entity.
    Takes one of the following values.
    - oper-status-up(1): the PCEP entity is active.
    - oper-status-down(2): the PCEP entity is inactive.
    - oper-status-going-up(3): the PCEP entity is activating.
    - oper-status-going-down(4): the PCEP entity is deactivating.
    - oper-status-failed(5): the PCEP entity has failed and will recover when possible.
    - oper-status-failed-perm(6): the PCEP entity has failed and will not recover without operator intervention."
}

uses info {
  description
    "Local PCEP entity information";
}

container pce-info {
  when ".../role = 'pce'"
    + "or "
    + ".../role = 'pcc-and-pce'" {
      description
        "These fields are applicable when the role is PCE.";
  }
  description
    "The Local PCE Entity PCE information";
  uses pce-info {
    description
      "Local PCE information";
  }

container path-key {
  if-feature "path-key"
  description
    "Path-Key Configuration";
  reference
    "RFC 5520: Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism";
  leaf enabled {
    type boolean;
    description
      "Enabled or Disabled";
  }
  leaf discard-timer {
type uint32;
  units "minutes";
  default "10";
  description
      "A timer to discard unwanted path-keys";
}
leaf reuse-time {
  type uint32;
  units "minutes";
  default "30";
  description
      "A time after which the path-keys could be reused";
}
leaf pce-id {
  type inet:ip-address;
  description
      "PCE Address to be used in each Path-Key Subobject (PKS)";
}
leaf connect-timer {
  type uint16 {
    range "1..65535";
  }
  units "seconds";
  default "60";
  description
      "The time in seconds that the PCEP entity will wait to establish a
TCP connection with a peer. If a TCP connection is not established
within this time then PCEP aborts the session setup attempt.";
  reference
      "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}
leaf connect-max-retry {
  type uint32;
  default "5";
  description
      "The maximum number of times the system tries to establish a TCP
connection to a peer before the session with the peer transitions to the idle
state.";
  reference
      "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}
leaf init-backoff-timer {
  type uint16 {
    range "1..65535";
  }
  units "seconds";
  description
    "The initial back-off time in seconds for retrying
     a failed session setup attempt to a peer.
     The back-off time increases for each failed
     session setup attempt, until a maximum back-off
     time is reached. The maximum back-off time is
     max-backoff-timer.";
  reference
    "RFC 5440: Path Computation Element (PCE) Communication
     Protocol (PCEP)";
}
leaf max-backoff-timer {
  type uint32;
  units "seconds";
  description
    "The maximum back-off time in seconds for retrying
     a failed session setup attempt to a peer.
     The back-off time increases for each failed session
     setup attempt, until this maximum value is reached.
     Session setup attempts then repeat periodically
     without any further increase in back-off time.";
  reference
    "RFC 5440: Path Computation Element (PCE) Communication
     Protocol (PCEP)";
}
leaf open-wait-timer {
  type uint16 {
    range "1..65535";
  }
  units "seconds";
  default "60";
  description
    "The time in seconds that the PCEP entity will wait
     to receive an Open message from a peer after the
     TCP connection has come up.
     If no Open message is received within this time then
     PCEP terminates the TCP connection and deletes the
     associated sessions.";
  reference
    "RFC 5440: Path Computation Element (PCE) Communication
     Protocol (PCEP)";
}
leaf keep-wait-timer {
type uint16 {
  range "1..65535";
} units "seconds";
default "60";

description
  "The time in seconds that the PCEP entity will wait
to receive a Keepalive or PCErr message from a peer
during session initialization after receiving an
Open message. If no Keepalive or PCErr message is
received within this time then PCEP terminates the
TCP connection and deletes the associated
sessions.";
reference
  "RFC 5440: Path Computation Element (PCE) Communication
  Protocol (PCE)";
}

leaf keep-alive-timer {
  type uint8;
  units "seconds";
  default "30";

description
  "The keep alive transmission timer that this PCEP
  entity will propose in the initial OPEN message of
each session it is involved in. This is the
  maximum time between two consecutive messages sent
to a peer. Zero means that the PCEP entity prefers
  not to send Keepalives at all.
  Note that the actual Keepalive transmission
  intervals, in either direction of an active PCEP
  session, are determined by negotiation between the
  peers as specified by RFC 5440, and so may differ
  from this configured value.";
reference
  "RFC 5440: Path Computation Element (PCE) Communication
  Protocol (PCE)";
}

leaf dead-timer {
  type uint8;
  units "seconds";
  must '(. > ../keep-alive-timer)' {
    error-message "The dead timer must be "
      + "larger than the keep alive timer";
  }
  default "120";

description
  "The dead timer that this PCEP entity will propose
  in the initial OPEN message of each session it is
involved in. This is the time after which a peer should declare a session down if it does not receive any PCEP messages. Zero suggests that the peer does not run a dead timer at all.

reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"

leaf allow-negotiation {
    type boolean;
description
    "Whether the PCEP entity will permit negotiation of session parameters."
}

leaf max-keep-alive-timer {
    type uint8;
    units "seconds";
description
    "In PCEP session parameter negotiation in seconds, the maximum value that this PCEP entity will accept from a peer for the interval between Keepalive transmissions. Zero means that the PCEP entity will allow no Keepalive transmission at all."
}

leaf max-dead-timer {
    type uint8;
    units "seconds";
description
    "In PCEP session parameter negotiation in seconds, the maximum value that this PCEP entity will accept from a peer for the Dead timer. Zero means that the PCEP entity will allow not running a Dead timer."
}

leaf min-keep-alive-timer {
    type uint8;
    units "seconds";
description
    "In PCEP session parameter negotiation in seconds, the minimum value that this PCEP entity will accept for the interval between Keepalive transmissions. Zero means that the PCEP entity insists on no Keepalive transmission at all."
}

leaf min-dead-timer {
    type uint8;
    units "seconds";
}
description
"In PCEP session parameter negotiation in
seconds, the minimum value that this PCEP entity
will accept for the Dead timer. Zero means that
the PCEP entity insists on not running a Dead
timer.";
}
leaf sync-timer {
    if-feature "svec";
    type uint16;
    units "seconds";
    default "60";
    description
    "The value of SyncTimer in seconds is used in the
case of synchronized path computation request
using the SVEC object. Consider the case where a
PCReq message is received by a PCE that contains
the SVEC object referring to M synchronized path
computation requests. If after the expiration of
the SyncTimer all the M path computation requests
have not been, received a protocol error is
triggered and the PCE MUST cancel the whole set
of path computation requests.
The aim of the SyncTimer is to avoid the storage
of unused synchronized requests should one of
them get lost for some reasons (for example, a
misbehaving PCC).
Zero means that the PCEP entity does not use the
SyncTimer.";
    reference
    "RFC 5440: Path Computation Element (PCE)
Communication Protocol (PCE)";
}
leaf request-timer {
    type uint16 {
        range "1..65535";
    }
    units "seconds";
    description
    "The maximum time that the PCEP entity will wait
for a response to a PCReq message.";
}
leaf max-sessions {
    type uint32;
    description
    "Maximum number of sessions involving this PCEP
entity that can exist at any time.";
}
leaf max-unknown-reqs {
    type uint32;
    default "5";
    description
    "The maximum number of unrecognized requests and
    replies that any session on this PCEP entity is
    willing to accept per minute before terminating
    the session.
    A PCRep message contains an unrecognized reply
    if it contains an RP object whose request ID
    does not correspond to any in-progress request
    sent by this PCEP entity.
    A PCReq message contains an unrecognized request
    if it contains an RP object whose request ID is
    zero.";
    reference
    "RFC 5440: Path Computation Element (PCE)
     Communication Protocol (PCE)";
}
leaf max-unknown-msgs {
    type uint32;
    default "5";
    description
    "The maximum number of unknown messages that any
    session on this PCEP entity is willing to accept
    per minute before terminating the session.";
    reference
    "RFC 5440: Path Computation Element (PCE)
     Communication Protocol (PCE)";
}
leaf pcep-notification-max-rate {
    type uint32;
    mandatory true;
    description
    "This variable indicates the maximum number of
    notifications issued per second. If events
    occur more rapidly, the implementation may
    simply fail to emit these notifications during
    that period, or may queue them until an
    appropriate time. A value of 0 means no
    notifications are emitted and all should be
    discarded (that is, not queued).";
}
container stateful-parameter {
    if-feature "stateful";
    description
    "The configured stateful PCE parameters";
    leaf state-timeout {
    ...
type uint32;
units "seconds";
description
"When a PCEP session is terminated, a PCC waits for this time period before flushing LSP state associated with that PCEP session and reverting to operator-defined default parameters or behaviours.";
}
leaf redelegation-timeout {
when "./.../role = 'pcc'"
+ "or"
+ "./.../role = 'pcc-and-pce'" {

description
"This field is applicable when the role is PCC";
}
type uint32;
units "seconds";
description
"When a PCEP session is terminated, a PCC waits for this time period before revoking LSP delegation to a PCE and attempting to redelegate LSPs associated with the terminated PCEP session to an alternate PCE.";
}
leaf rpt-non-pcep-lsp {
when "./.../role = 'pcc'"
+ "or"
+ "./.../role = 'pcc-and-pce'" {

description
"This field is applicable when the role is PCC";
}
type boolean;
default "true";
description
"If set, a PCC reports LSPs that are not controlled by any PCE (for example, LSPs that are statically configured at the PCC). ";
}
reference
"RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}
when "./.role = 'pce'"
+ "or "
+ "./.role = 'pcc-and-pce'" {
    description
    "These field are applicable when the role is PCE";
}

if-feature "objective-function";
uses of-list;
description
    "The authorized OF-List at PCE for all peers";
}

container lsp-db {
    if-feature "stateful";
    config false;
description
    "The LSP-DB";

    leaf db-ver {
        when "../../role = 'pcc'"
        + "or "
        + "../../role = 'pcc-and-pce'" {
            description
            "This field is applicable when the role is PCC";
        }
        if-feature "sync-opt";
type uint64;
description
    "The LSP State Database Version Number";
}

list association-list {
    if-feature "association";
    key "type id source global-source extended-id";
description
    "List of all PCEP associations";

    reference
    "I-D.ietf-pce-association-group: PCEP Extensions for Establishing Relationships Between Sets of LSPs";

    leaf type {
        type identityref {
            base te-types:association-type;
        }
description
    "The PCEP association type";
    }

    leaf id {
        type uint16;
    }


description
  "PCEP Association ID";
}
leaf source {
  type inet:ip-address;
  description
    "PCEP Association Source.";
}
leaf global-source {
  type uint32;
  description
    "PCEP Association Global Source.";
}
leaf extended-id {
  type string;
  description
    "Additional information to support unique identification.";
}
list lsp {
  key "plsp-id pcc-id lsp-id";
  description
    "List of all LSP in this association";
  leaf plsp-id {
    type leafref {
      path "/pcep/entity/lsp-db/" + "lsp/plsp-id";
    }
    description
      "Reference to PLSP-ID in LSP-DB";
  }
  leaf pcc-id {
    type leafref {
      path "/pcep/entity/lsp-db/" + "lsp/pcc-id";
    }
    description
      "Reference to PCC-ID in LSP-DB";
  }
  leaf lsp-id {
    type leafref {
      path "/pcep/entity/lsp-db/" + "lsp/lsp-id";
    }
    description
      "Reference to LSP ID in LSP-DB";
  }
}
list lsp {
  key "plsp-id pcc-id lsp-id";
  description "List of all LSPs in LSP-DB";
  leaf plsp-id {
    type uint32 {
      range "1..1048575";
    }
    description "A PCEP-specific identifier for the LSP. A PCC creates a unique PLSP-ID for each LSP that is constant for the lifetime of a PCEP session. PLSP-ID is 20 bits with 0 and 0xFFFFF are reserved";
  }
  leaf pcc-id {
    type inet:ip-address;
    description "The local internet address of the PCC, that generated the PLSP-ID.";
  }
  leaf source {
    type leafref {
    }
    description "Tunnel sender address extracted from SENDER_TEMPLATE object";
    reference "RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels";
  }
  leaf destination {
    type leafref {
      path "/te:te/te:lsps-state/te:lsp/te:" + "destination";
    }
    description "Tunnel endpoint address extracted from SESSION object";
    reference "RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels";
  }
  leaf tunnel-id {
    type leafref {
      path "/te:te/te:lsps-state/te:lsp/te:tunnel-id";
    }
  }
}

description
"LSP is delegated or not";
}
leaf peer {
  type leafref {
    path "/pcep/entity/peers/peer/addr";
  }
  must './../enabled = true()' {
    error-message "The LSP must be delegated";
  }
  description
  "At the PCC, the reference to the PCEP peer to
  which LSP is delegated; At the PCE, the
  reference to the PCEP peer which delegated this
  LSP";
}
leaf srp-id {
  type uint32;
  description
  "The last SRP-ID-number associated with this
  LSP.";
}
}
container initiation {
  if-feature "pce-initiated";
  description
  "The PCE initiation related parameters";
  reference
  "RFC 8281: Path Computation Element Communication
  Protocol (PCEP) Extensions for PCE-Initiated LSP
  Setup in a Stateful PCE Model";
  leaf enabled {
    type boolean;
    description
    "Set to true if this LSP is initiated by a PCE";
  }
  leaf peer {
    type leafref {
      path "/pcep/entity/peers/peer/addr";
    }
    must './../enabled = true()' {
      error-message "The LSP must be PCE-Initiated";
    }
    description
    "At the PCC, the reference to the PCEP peer
    that initiated this LSP; At the PCE, the
    reference to the PCEP peer where the LSP
    is initiated";
leaf symbolic-path-name {
  type string;
  description "The symbolic path name associated with the LSP.";
  reference "RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
}

leaf last-error {
  type identityref {
    base lsp-error;
  }
  description "The last error for the LSP.";
}

leaf pst {
  type identityref {
    base te-types:path-signaling-type;
  }
  default "te-types:path-setup-rsvp";
  description "The Path Setup Type";
  reference "RFC 8408: Conveying Path Setup Type in PCE Communication Protocol (PCEP) Messages";
}

list association-list {
  if-feature "association";
  key "type id source global-source extended-id";
  description "List of all PCEP associations";
  leaf type {
    type leafref {
      path "/pcep/entity/lsp-db/association-list/type";
    }
    description "PCEP Association Type";
  }
  leaf id {
    type leafref {
      path "/pcep/entity/lsp-db/association-list/id";
    }
    description "PCEP Association ID";
  }
}
leaf source {
    type leafref {
        path "/pcep/entity/lsp-db/"
        + "association-list/source";
    }
    description
        "PCEP Association Source.";
}
leaf global-source {
    type leafref {
        path "/pcep/entity/lsp-db/"
        + "association-list/global-source";
    }
    description
        "PCEP Association Global Source.";
}
leaf extended-id {
    type leafref {
        path "/pcep/entity/lsp-db/"
        + "association-list/extended-id";
    }
    description
        "Additional information to support unique identification.";
}
reference
    "I-D.ietf-pce-association-group: PCEP
     Extensions for Establishing Relationships Between Sets of LSPs";
}
}
container path-keys {
    when ".../role = 'pce' or ../role = 'pcc-and-pce'" {
        description
            "These fields are applicable when the role is PCE";
    }
    if-feature "path-key"
        config false;
    description
        "The path-keys generated by the PCE";
    reference
        "RFC 5520: Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism";
    list path-keys {

key "path-key";
description
  "The list of path-keys generated by the PCE";
leaf path-key {
  type uint16;
  description
    "The identifier, or token used to represent
     the Confidential Path Segment (CPS) within
     the context of the PCE";
}
container cps {
  description
    "The Confidential Path Segment (CPS)";
  list explicit-route-objects {
    key "index";
    description
      "List of explicit route objects";
    leaf index {
      type uint32;
      description
        "ERO subobject index";
    }
    uses te-types:explicit-route-hop;
  }
}
leaf pcc-original {
  type leafref {
    path "/pcep/entity/peers/peer/addr";
  }
  description
    "Reference to PCC peer address of
     the original request";
}
leaf req-id {
  type uint32;
  description
    "The request ID of the original PCReq.";
}
leaf retrieved {
  type boolean;
  description
    "If path-key has been retrieved yet";
}
leaf pcc-retrieved {
  type leafref {
    path "/pcep/entity/peers/peer/addr";
  }
  must '(../retrieved = true())' 
}
error-message "The Path-key should be retrieved";
}

leaf creation-time {
  type yang:timestamp;
  description
      "The timestamp value at the time this Path-Key was created.";
}

leaf discard-time {
  type uint32;
  units "minutes";
  description
      "A time after which this path-keys will be discarded";
}

leaf reuse-time {
  type uint32;
  units "minutes";
  description
      "A time after which this path-keys could be reused";
}


container peers {
  description
      "The list of configured peers for the entity (remote PCE)";
  list peer {
    key "addr";
    description
      "The peer configured for the entity. (remote PCE)";
    leaf addr {
      type inet:ip-address;
      description
        "The local Internet address of this PCEP peer.";
    }
    leaf role {
      type pcep-role;
      mandatory true;
      description
        "The role of the PCEP Peer."
    }
  }
}
Takes one of the following values.
- unknown(0): this PCEP peer role is not known.
- pcc(1): this PCEP peer is a PCC.
- pce(2): this PCEP peer is a PCE.
- pcc-and-pce(3): this PCEP peer is both a PCC and a PCE.

leaf description {
  type string;
  description
    "Description of the PCEP peer configured by the user";
}

uses info {
  description
    "PCE Peer information";
}

container pce-info {
  uses pce-info {
    description
      "PCE Peer information";
  }
  description
    "The PCE Peer information";
}

leaf delegation-pref {
  if-feature "stateful";
  type uint8 {
    range "0..7";
  }
  description
    "The PCE peer delegation preference.";
}

container auth {
  description
    "The Authentication options";
  choice auth-type-selection {
    description
      "Options for expressing authentication setting.";
    case auth-key-chain {
      leaf key-chain {
        type key-chain:key-chain-ref;
        description
          "key-chain name.";
      }
    }
  }
}
case auth-key {
  leaf crypto-algorithm {
    type identityref {
      base key-chain:crypto-algorithm;
    }
    mandatory true;
    description "Cryptographic algorithm associated with key.";
  }
  choice key-string-style {
    description "Key string styles";
    case keystring {
      leaf keystring {
        nacm:default-deny-all;
        type string;
        description "Key string in ASCII format.";
      }
    }
    case hexadecimal {
      if-feature "key-chain:hex-key-string";
      leaf hexadecimal-string {
        nacm:default-deny-all;
        type yang:hex-string;
        description "Key in hexadecimal string format. When compared to ASCII, specification in hexadecimal affords greater key entropy with the same number of internal key-string octets. Additionally, it discourages usage of well-known words or numbers.";
      }
    }
  }
}

case auth-tls {
  if-feature "tls";
  choice role {
    description "The role of the local entity";
    case server {
      container tls-server {
        uses tlss:tls-server-grouping {
          description "Server TLS information.";
        }
      }
    }
  }
}

description "Client TLS information.";
}

case client {
    container tls-client {
        uses tlsc:tls-client-grouping {
            description "Client TLS information.";
            description "TLS related information";
        }
    }
}

leaf discontinuity-time {
    type yang:timestamp;
    config false;
    description "The timestamp of the time when the information and
    statistics were last reset.";
}

leaf initiate-session {
    type boolean;
    config false;
    description "Indicates whether the local PCEP entity initiates
    sessions to this peer, or waits for the peer to
    initiate a session.";
}

leaf session-exists {
    type boolean;
    config false;
    description "Indicates whether a session with
    this peer currently exists.";
}

leaf session-up-time {
    type yang:timestamp;
    config false;
    description "The timestamp value of the last time a
    session with this peer was successfully
    established.";
}
leaf session-fail-time {
  type yang:timestamp;
  config false;
  description
    "The timestamp value of the last time a session with this peer failed to be established.";
}

leaf session-fail-up-time {
  type yang:timestamp;
  config false;
  description
    "The timestamp value of the last time a session with this peer failed from active.";
}

container sessions {
  config false;
  description
    "This entry represents a single PCEP session in which the local PCEP entity participates. This entry exists only if the corresponding PCEP session has been initialized by some event, such as manual user configuration, auto-discovery of a peer, or an incoming TCP connection.";
  list session {
    key "initiator";
    description
      "The list of sessions, note that for a time being two sessions may exist for a peer";
    leaf initiator {
      type pcep-initiator;
      description
        "The initiator of the session, that is, whether the TCP connection was initiated by the local PCEP entity or the peer. There is a window during session initialization where two sessions can exist between a pair of PCEP speakers, each initiated by one of the speakers. One of these sessions is always discarded before it leaves OpenWait state. However, before it is discarded, two sessions to the given peer appear transiently in this yang module. The sessions are distinguished by who initiated them, and so this field is the key.";
    }
  }
}
leaf role {
  type leafref {
    path "/pcep/entity/role";
  }
  description "The reference to peer role.";
}
leaf state-last-change {
  type yang:timestamp;
  description "The timestamp value at the time this
  session entered its current state as
denoted by the state leaf.";
}
leaf state {
  type pcep-sess-state;
  description "The current state of the session.
The set of possible states excludes the
idle state since entries do not exist
in the idle state.";
}
leaf session-creation {
  type yang:timestamp;
  description "The timestamp value at the time this
session was created.";
}
leaf connect-retry {
  type yang:counter32;
  description "The number of times that the local PCEP
entity has attempted to establish a TCP
connection for this session without
success. The PCEP entity gives up when
this reaches connect-max-retry.";
}
leaf local-id {
  type uint8;
  description "The value of the PCEP session ID used by
the local PCEP entity in the Open message
for this session. If state is tcp-pending
then this is the session ID that will be
used in the Open message. Otherwise, this
is the session ID that was sent in the
Open message.";
  reference
}
leaf remote-id {
  type uint8;
  must "((../state != 'tcp-pending'
    + "and 
    + "../state != 'open-wait' )
    + "or 
    + "((../state = 'tcp-pending'
      + " or 
      + "../state = 'open-wait' )
      + "and (. = 0))" 
    error-message "Invalid remote-id";
  description
    "If state is tcp-pending or open-wait then
     this leaf is not used and MUST be set to
     zero."; 
}

description
  "The value of the PCEP session ID used by the
  peer in its Open message for this session.";
reference
  "RFC 5440: Path Computation Element (PCE)
  Communication Protocol (PCE)";

leaf keepalive-timer {
  type uint8;
  units "seconds";
  must "((../state = 'session-up'
    + "or 
    + "((../state != 'session-up'
      + "and (. = 0))" 
    error-message "Invalid keepalive timer";
  description
    "This field is used if and only if state is
     session-up. Otherwise, it is not used and
     MUST be set to zero.";
}

description
  "The agreed maximum interval at which the local
  PCEP entity transmits PCEP messages on this PCEP
  session. Zero means that the local PCEP entity
  never sends Keepalives on this session.";
reference
  "RFC 5440: Path Computation Element (PCE)
  Communication Protocol (PCE)";
leaf peer-keepalive-timer {
    type uint8;
    units "seconds";
    must "((../state = 'session-up'"
        + " or "
        + "../state != 'session-up'"
        + " and "
        + "(. = 0)))" {
        error-message "Invalid Peer keepalive timer";
        description
            "This field is used if and only if state is session-up. Otherwise, it is not used and MUST be set to zero."
    }
    description
        "The agreed maximum interval at which the peer transmits PCEP messages on this PCEP session. Zero means that the peer never sends Keepalives on this session."
    reference
        "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
}
leaf dead-timer {
    type uint8;
    units "seconds";
    description
        "The dead timer interval for this PCEP session."
    reference
        "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
}
leaf peer-dead-timer {
    type uint8;
    units "seconds";
    must "((../state != 'tcp-pending'
        + " and "
        + "../state != 'open-wait'"
        + " or "
        + "((../state = 'tcp-pending'
            + " or "
            + "../state = 'open-wait'
                + " and "
                + "(. = 0)))" {
        error-message "Invalid Peer Dead timer";
        description
            "If state is tcp-pending or open-wait then this leaf is not used and MUST be set to zero."
    }
leaf ka-hold-time-rem {
  type uint8;
  units "seconds";
  must "((../state != 'tcp-pending'
    + "and "
    + "./state != 'open-wait' ) 
    + "or "
    + "((../state = 'tcp-pending'
    + "or "
    + "./state = 'open-wait' )
    + "and "
    + "(. = 0)) )") {
    error-message
    "Invalid Keepalive hold time remaining";
    description
    "If state is tcp-pending or open-wait then this
    field is not used and MUST be set to zero.";
  }
  description
  "The keep alive hold time remaining for this
  session.";
}
leaf overloaded {
  type boolean;
  description
  "If the local PCEP entity has informed the peer that
  it is currently overloaded, then this is set to true.
  Otherwise, it is set to false.";
  reference
  "RFC 5440: Path Computation Element (PCE)
  Communication Protocol (PCEP)"
}
leaf overload-time {
  type uint32;
  units "seconds";
  must 
  "((../overloaded = true())
  + 'or ((../overloaded != true())
  + 'and (. = 0)))') {
    error-message "Invalid overload-time";
    description
  }
"This field is only used if overloaded is set to true. Otherwise, it is not used and MUST be set to zero."

leaf peer-overloaded {
  type boolean;
  description "If the peer has informed the local PCEP entity that it is currently overloaded, then this is set to true. Otherwise, it is set to false.";
  reference "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";
}

leaf peer-overload-time {
  type uint32;
  units "seconds";
  must '(((../peer-overloaded = ' + 'true()') or '
  '(((../peer-overloaded !=' + 'true())'
  + ' and '
  + '(. = 0)))')' {
    error-message "Invalid peer overload time";
    description "This field is only used if peer-overloaded is set to true. Otherwise, it is not used and MUST be set to zero.";
  }
  description "The interval of time that is remaining until the peer will cease to be overloaded. If it is not known how long the peer will stay in overloaded state, this leaf is set to zero.";
  reference "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";
}

leaf lspdb-sync {
  if-feature "stateful";
type sync-state;
    description
        "The LSP-DB state synchronization status.";
    reference
        "RFC 8231: Path Computation Element Communication
         Protocol (PCEP) Extensions for Stateful PCE";

leaf recv-db-ver {
    when "../role = 'pcc'
        or "
        ../role = 'pcc-and-pce'
    {
        description
            "This field is applicable when the role is PCC";
    }
    if-feature "stateful";
    if-feature "sync-opt";
    type uint64;
    description
        "The last received LSP State Database Version Number";
    reference
        "RFC 8231: Path Computation Element Communication
         Protocol (PCEP) Extensions for Stateful PCE";
}

container of-list {
    when "../role = 'pce'
        or "
        ../role = 'pcc-and-pce'
    {
        description
            "These fields are applicable when the role is PCE";
    }
    if-feature "objective-function";
    uses of-list;
    description
        "Indicate the list of supported OF on this session";
    reference
        "RFC 5541: Encoding of Objective Functions in
         the Path Computation Element Communication
         Protocol (PCEP)";
}

container pst-list {
    when "../role = 'pce'
        or "
        ../role = 'pcc-and-pce'
    {
        description
            "";
    }
}
"These fields are applicable when the role is PCE";
}
description
 "Indicate the list of supported PST on this session";
reference
 "RFC 8408: Conveying Path Setup Type in PCE Communication Protocol (PCEP) Messages";
list path-setup-type {
 key "pst";
 description
 "The list of authorized PST";
 leaf pst {
 type identityref {
   base te-types:path-signaling-type;
 }
 description
 "The PST authorized";
 }
}
container assoc-type-list {
 if-feature "association";
 description
 "Indicate the list of supported association types on this session";
 reference
 "I-D.ietf-pce-association-group: PCEP Extensions for Establishing Relationships Between Sets of LSPs";
 list assoc-type {
 key "at";
 description
 "The list of authorized association types";
 leaf at {
 type identityref {
   base te-types:association-type;
 }
 description
 "The association type authorized";
 }
}
leaf speaker-entity-id {
 if-feature "sync-opt";
 type string;
 description

"The Speaker Entity Identifier";
reference
"RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
}
leaf discontinuity-time {
  type yang:timestamp;
  description
    "The timestamp value of the time when the statistics were last reset.";
}
}
// session
}
// sessions
}
// peer
}
// peers
}
// entity
}

//pcep
/*
* Notifications
*/

notification pcep-session-up {
  description
    "This notification is sent when the value of '/pcep/peers/peer/sessions/session/state'
    enters the 'session-up' state."
  uses notification-instance-hdr;
  uses notification-session-hdr;
  leaf state-last-change {
    type yang:timestamp;
    description
      "The timestamp value at the time this session entered its current state as denoted by the state leaf.";
  }
  leaf state {
    type pcep-sess-state;
    description
      "The current state of the session. The set of possible states excludes the idle state
since entries do not exist in the idle state.;
}
reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";
}

//notification
notification pcep-session-down {
description
"This notification is sent when the value of
'/pcep/peers/peer/sessions/session/state' leaves the 'session-up' state.";
uses notification-instance-hdr;
leaf session-initiator {
type pcep-initiator;
description
"The initiator of the session.";
}
leaf state-last-change {
type yang:timestamp;
description
"The timestamp value at the time this session entered its current state as denoted by the state leaf.";
}
leaf state {
type pcep-sess-state;
description
"The current state of the session. The set of possible states excludes the idle state since entries do not exist in the idle state.";
}
reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)";
}

//notification
notification pcep-session-local-overload {
description
"This notification is sent when the local PCEP entity enters overload state for a peer.";
uses notification-instance-hdr;
uses notification-session-hdr;
leaf overloaded {
type boolean;
description
  "If the local PCEP entity has informed the peer
  that it is currently overloaded, then this is set
to true. Otherwise, it is set to false.";
}
leaf overload-time {
  type uint32;
  units "seconds";
  description
  "The interval of time that is remaining until the
  local PCEP entity will cease to be overloaded on
  this session.";
}
reference
  "RFC 5440: Path Computation Element (PCE) Communication
  Protocol (PCEP)";
}

//notification
notification pcep-session-local-overload-clear {
  description
  "This notification is sent when the local PCEP entity
  leaves overload state for a peer.";
  uses notification-instance-hdr;
  leaf overloaded {
    type boolean;
    description
    "If the local PCEP entity has informed the peer
    that it is currently overloaded, then this is set
to true. Otherwise, it is set to false.";
  }
  reference
  "RFC 5440: Path Computation Element (PCE) Communication
  Protocol (PCEP)";
}

//notification
notification pcep-session-peer-overload {
  description
  "This notification is sent when a peer enters overload
  state.";
  uses notification-instance-hdr;
  uses notification-session-hdr;
  leaf peer-overloaded {
    type boolean;
description
"If the peer has informed the local PCEP entity that it is currently overloaded, then this is set to true. Otherwise, it is set to false."
}
leaf peer-overload-time {
  type uint32;
  units "seconds";
  description
  "The interval of time that is remaining until the peer will cease to be overloaded. If it is not known how long the peer will stay in overloaded state, this leaf is set to zero."
}
reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"
}
//notification
notification pcep-session-peer-overload-clear {
  description
  "This notification is sent when a peer leaves overload state."
  uses notification-instance-hdr;
  leaf peer-overloaded {
    type boolean;
    description
    "If the peer has informed the local PCEP entity that it is currently overloaded, then this is set to true. Otherwise, it is set to false."
  }
  reference
  "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"
}
//notification
/*
 * RPC
 */
rpc trigger-resync {
  if-feature "stateful";
  if-feature "sync-opt";
  description
  "Trigger the resyncrization at the PCE";
reference
"RFC 8232: Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE";
input {
  leaf pcc {
    type leafref {
      path "/pcep/entity/peers/peer/addr";
    }
    description
    "The IP address to identify the PCC. The state synchronization is re-triggered for all LSPs from the PCC. The rpc on the PCC will be ignored.";
  }
}
//rpc

<CODE ENDS>

10.2. ietf-pcep-stats module

<CODE BEGINS> file "ietf-pcep-stats@2019-07-01.yang"
module ietf-pcep-stats {
  yang-version 1.1;
  prefix pcep-stats;
  import ietf-pcep {
    prefix pcep;
    reference
    "RFC XXXX: A YANG Data Model for Path Computation Element Communications Protocol (PCEP)";
  }
  import ietf-yang-types {
    prefix yang;
    reference
    "RFC 6991: Common YANG Data Types";
  }
  organization
  "IETF PCE (Path Computation Element) Working Group";

The YANG module augments the PCEP yang operational model with statistics, counters and telemetry data.

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.

revision 2019-07-01 {
  description
    "Initial revision.";
  reference
    "RFC XXXX:  A YANG Data Model for Path Computation Element Communications Protocol (PCEP)";
}

/*
 * Groupings
 */

grouping pcep-stats {
  description
    "This grouping defines statistics for PCEP. It is used for both peer and current session.";
  leaf avg-rsp-time {
    when "../../pcep:role = 'pce'"
    + "or"
    + "../../pcep:role = 'pcc-and-pce'" {
      description
        "Valid for PCEP Peer as PCE";
    } type uint32;
    units "milliseconds";
    description
  }
}

"The average response time. If an average response time has not been calculated then this leaf has the value zero."

} leaf lwm-rsp-time {
   when "./../pcep:role = 'pce'
   + "or"
   + "./../pcep:role = 'pcc-and-pce'"
   description
   "Valid for PCEP Peer as PCE"
} type uint32;
units "milliseconds"

description
"The smallest (low-water mark) response time seen. If no responses have been received then this leaf has the value zero."

} leaf hwm-rsp-time {
   when "./../pcep:role = 'pce'
   + "or"
   + "./../pcep:role = 'pcc-and-pce'"
   description
   "Valid for PCEP Peer as PCE"
} type uint32;
units "milliseconds"

description
"The greatest (high-water mark) response time seen. If no responses have been received then this object has the value zero."

} leaf num-pcreq-sent {
   when "./../pcep:role = 'pce'
   + "or"
   + "./../pcep:role = 'pcc-and-pce'"
   description
   "Valid for PCEP Peer as PCE"
} type yang:counter32;

description
"The number of PCReq messages sent."

} leaf num-pcreq-rcvd {
   when "./../pcep:role = 'pcc'
   + "or"
   + "./../pcep:role = 'pcc-and-pce'"
   description

leaf num-pcreq-received {
  when "../../pcep:role = 'pce'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
  + "Valid for PCEP Peer as PCC"
  type yang:counter32;
  description
  "The number of PCReq messages received.";
}

leaf num-pcrep-sent {
  when "././pcep:role = 'pcc'"
  + "or "
  + "././pcep:role = 'pcc-and-pce'"
  + "Valid for PCEP Peer as PCC"
  type yang:counter32;
  description
  "The number of PCRep messages sent.";
}

leaf num-pcrep-received {
  when "../../pcep:role = 'pce'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
  + "Valid for PCEP Peer as PCE"
  type yang:counter32;
  description
  "The number of PCRep messages received.";
}

leaf num-pcerr-sent {
  type yang:counter32;
  description
  "The number of PCErr messages sent.";
}

leaf num-pcerr-received {
  type yang:counter32;
  description
  "The number of PCErr messages received.";
}

leaf num-pcntf-sent {
  type yang:counter32;
  description
  "The number of PCNtf messages sent.";
}

leaf num-pcntf-received {
  type yang:counter32;
  description
  "The number of PCNtf messages received.";
}
leaf num-keepalive-sent {
  type yang:counter32;
  description
      "The number of Keepalive messages sent.";
}
leaf num-keepalive-rcvd {
  type yang:counter32;
  description
      "The number of Keepalive messages received.";
}
leaf num-unknown-rcvd {
  type yang:counter32;
  description
      "The number of unknown messages received.";
}
leaf num-corrupt-rcvd {
  type yang:counter32;
  description
      "The number of corrupted PCEP message received.";
}
leaf num-req-sent {
  when ":../../pcep:role = 'pce'
      + "or "
      + ":../../pcep:role = 'pcc-and-pce"
    { 
      description
        "Valid for PCEP Peer as PCE";
    }
  type yang:counter32;
  description
      "The number of requests sent. A request corresponds
       1:1 with an RP object in a PCReq message. This might
       be greater than num-pcreq-sent because multiple
       requests can be batched into a single PCReq
       message.";
}
leaf num-req-sent-pend-rep {
  when ":../../pcep:role = 'pce'
      + "or "
      + ":../../pcep:role = 'pcc-and-pce"
    { 
      description
        "Valid for PCEP Peer as PCE";
    }
  type yang:counter32;
  description
      "The number of requests that have been sent for
       which a response is still pending.";
}
leaf num-req-sent-ero-rcvd {
when "./.../pcep:role = 'pce'"
+ "or "
+ "./.../pcep:role = 'pcc-and-pce'" 
  description
  "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of requests that have been sent for which a response with an ERO object was received. Such responses indicate that a path was successfully computed by the peer.";
}
leaf num-req-sent-nopath-rcvd {
  when "./.../pcep:role = 'pce'"
  + "or "
  + "./.../pcep:role = 'pcc-and-pce'" 
  description
  "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of requests that have been sent for which a response with a NO-PATH object was received. Such responses indicate that the peer could not find a path to satisfy the request.";
}
leaf num-req-sent-cancel-rcvd {
  when "./.../pcep:role = 'pce'"
  + "or "
  + "./.../pcep:role = 'pcc-and-pce'" 
  description
  "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of requests that were cancelled with a PCNtf message. This might be different than num-pcntf-rcvd because not all PCNtf messages are used to cancel requests, and a single PCNtf message can cancel multiple requests.";
}
leaf num-req-sent-error-rcvd {
  when "./.../pcep:role = 'pce'"
  + "or "
  + "./.../pcep:role = 'pcc-and-pce'" 
  description
"
"Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of requests that were rejected with a
PCErr message. This might be different than
num-pcerr-rcvd because not all PCErr messages are
used to reject requests, and a single PCErr message
can reject multiple requests."
}
leaf num-req-sent-timeout {
  when "./../pcep:role = 'pce'
    + "or"
    + "./../pcep:role = 'pcc-and-pce'"
    { description
      "Valid for PCEP Peer as PCE";
    }
}
leaf num-req-sent-cancel-sent {
  when "./../pcep:role = 'pce'
    + "or"
    + "./../pcep:role = 'pcc-and-pce'"
    { description
      "Valid for PCEP Peer as PCE";
    }
}
leaf num-req-rcvd {
  when "./../pcep:role = 'pcc'
    + "or"
    + "./../pcep:role = 'pcc-and-pce'"
    { description
      "Valid for PCEP Peer as PCC";
    }
}
type yang:counter32;
description
"The number of requests received. A request
  corresponds 1:1 with an RP object in a PCReq
  message.

This might be greater than num-pcreq-rcvd because multiple requests can be batched into a single PCReq message.

leaf num-req-rcvd-pend-rep {
  when "../../pcep:role = 'pcc'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests that have been received for which a response is still pending."

leaf num-req-rcvd-ero-sent {
  when "../../pcep:role = 'pcc'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests that have been received for which a response with an ERO object was sent. Such responses indicate that a path was successfully computed by the local PCEP entity."

leaf num-req-rcvd-nopath-sent {
  when "../../pcep:role = 'pcc'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests that have been received for which a response with a NO-PATH object was sent. Such responses indicate that the local PCEP entity could not find a path to satisfy the request."

leaf num-req-rcvd-cancel-sent {
  when "../../pcep:role = 'pcc'"
  + "or "
  + "../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests that have been received for which a response with a CANCEL object was sent. Such responses indicate that the local PCEP entity could not find a path to satisfy the request."

description
"Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests received that were cancelled
by the local PCEP entity sending a PCNtf message.
This might be different than num-pcntf-sent because
not all PCNtf messages are used to cancel requests,
and a single PCNtf message can cancel multiple
requests.";
}
leaf num-req-rcvd-error-sent {
  when "././pcep:role = 'pcc'
    + "or "
    + "././pcep:role = 'pcc-and-pce'
  {
    description
    "Valid for PCEP Peer as PCC";
  }
type yang:counter32;
description
"The number of requests received that were cancelled
by the local PCEP entity sending a PCErr message.
This might be different than num-pcerr-sent because
not all PCErr messages are used to cancel requests,
and a single PCErr message can cancel multiple
requests.";
}
leaf num-req-rcvd-cancel-rcvd {
  when "././pcep:role = 'pcc'
    + "or "
    + "././pcep:role = 'pcc-and-pce'
  {
    description
    "Valid for PCEP Peer as PCC";
  }
type yang:counter32;
description
"The number of requests that were received from the
peer and explicitly cancelled by the peer sending
a PCNtf.";
}
leaf num-rep-rcvd-unknown {
  when "././pcep:role = 'pce'
    + "or "
    + "././pcep:role = 'pcc-and-pce'
  {
    description
    "Valid for PCEP Peer as PCE";
  }
type yang:counter32;
description
"The number of responses to unknown requests received. A response to an unknown request is a response whose RP object does not contain the request ID of any request that is currently outstanding on the session.";
}
leaf num-req-rcvd-unknown {
  when "../../pcep:role = 'pcc'"
  + "or "
  + "../../../pcep:role = 'pcc-and-pce'"
  { description
    "Valid for PCEP Peer as PCC";
  }
  type yang:counter32;
description
"The number of unknown requests that have been received. An unknown request is a request whose RP object contains a request ID of zero.";
}
container svec {
  if-feature "pcep:svec";
description
"If synchronized path computation is supported";
leaf num-svec-sent {
  when "../../../pcep:role = 'pce'"
  + "or "
  + "../../../pcep:role = 'pcc-and-pce'"
  { description
    "Valid for PCEP Peer as PCE";
  }
  type yang:counter32;
description
"The number of SVEC objects sent in PCReq messages. An SVEC object represents a set of synchronized requests.";
}
leaf num-svec-req-sent {
  when "../../../pcep:role = 'pce'"
  + "or "
  + "../../../pcep:role = 'pcc-and-pce'"
  { description
    "Valid for PCEP Peer as PCE";
  }
  type yang:counter32;
description
"The number of requests sent that appeared in one
or more SVEC objects.;
}
leaf num-svec-rcvd {
  when "../../../pcep:role = 'pcc'"
    + "or "
    + "../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC";
  }
  type yang:counter32;
  description
  "The number of SVEC objects received in PCReq
   messages. An SVEC object represents a set of
   synchronized requests.";
}
leaf num-svec-req-rcvd {
  when "../../../pcep:role = 'pcc'"
    + "or "
    + "../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC";
  }
  type yang:counter32;
  description
  "The number of requests received that appeared
   in one or more SVEC objects.";
}
}/svec
container stateful {
  if-feature "pcep:stateful";
  description
  "Stateful PCE related statistics";
  leaf num-pcrpt-sent {
    when "../../../pcep:role = 'pce'"
      + "or "
      + "../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCE";
    }
    type yang:counter32;
    description
    "The number of PCRpt messages sent.";
  }
  leaf num-pcrpt-rcvd {
    when "../../../pcep:role = 'pcc'"
      + "or "
      + "../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC";
    }
    type yang:counter32;
    description
    "The number of PCRpt messages received.";
  }
}
leaf num-pcupd-sent {
  when "../../../pcep:role = 'pcc'"
    + "or"
    + "../../../pcep:role = 'pcc-and-pce'"
  description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
  "The number of PCUpd messages sent.";
}
leaf num-pcupd-rcvd {
  when "../../../pcep:role = 'pce'"
    + "or"
    + "../../../pcep:role = 'pcc-and-pce'"
  description
    "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
  "The number of PCUpd messages received.";
}
leaf num-rpt-sent {
  when "../../../pcep:role = 'pce'"
    + "or"
    + "../../../pcep:role = 'pcc-and-pce'"
  description
    "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
  "The number of LSP Reports sent. A LSP report corresponds 1:1 with an LSP object in a PCRpt message. This might be greater than num-pcrpt-sent because multiple reports can be batched into a single PCRpt message.";
}
leaf num-rpt-rcvd {
  when "../../../pcep:role = 'pcc'"
    + "or"
    + "../../../pcep:role = 'pcc-and-pce'"
  description
    "Valid for PCEP Peer as PCC";
}
description
"Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of LSP Reports received. A LSP report corresponds 1:1 with an LSP object in a PCRpt message. This might be greater than num-pcrpt-rcvd because multiple reports can be batched into a single PCRpt message."
}
leaf num-rpt-rcvd-error-sent {
  when ".../../../pcep:role = 'pcc'
    + "or "
    + ".../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC";
  }
type yang:counter32;
description
"The number of reports of LSPs received that were responded by the local PCEP entity by sending a PCErr message."
}
leaf num-upd-sent {
  when ".../../../pcep:role = 'pcc'
    + "or "
    + ".../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC";
  }
type yang:counter32;
description
"The number of LSP updates sent. A LSP update corresponds 1:1 with an LSP object in a PCUpd message. This might be greater than num-pcupd-sent because multiple updates can be batched into a single PCUpd message."
}
leaf num-upd-rcvd {
  when ".../../../pcep:role = 'pce'
    + "or "
    + ".../../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCE";
  }
type yang:counter32;
description
"The number of LSP Updates received. A LSP update
corresponds 1:1 with an LSP object in a PCUpd
message.
This might be greater than num-pcupd-rcvd because
multiple updates can be batched into a single
PCUpd message.";
}
leaf num-upd-rcvd-unknown {
when "../../../pcep:role = 'pce'
+ "or 
+ "../../../pcep:role = 'pcc-and-pce'"
{ 
description
"Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of updates to unknown LSPs
received. An update to an unknown LSP is a
update whose LSP object does not contain the
PLSP-ID of any LSP that is currently
present.";
}
leaf num-upd-rcvd-undelegated {
when "../../../pcep:role = 'pce'
+ "or 
+ "../../../pcep:role = 'pcc-and-pce'
{ 
description
"Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of updates to not delegated LSPs
received. An update to an undelegated LSP is a
update whose LSP object does not contain the
PLSP-ID of any LSP that is currently
delegated to current PCEP session.";
}
leaf num-upd-rcvd-error-sent {
when "../../../pcep:role = 'pce'
+ "or 
+ "../../../pcep:role = 'pcc-and-pce'
{ 
description
"Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of updates to LSPs received that were
responded by the local PCEP entity by sending a
PCErr message.
}
}
}

}
leaf num-initiate-rcvd-error-sent {
  when "../../pcep:role = 'pce'"
  + "or"
  + "../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCE"
  } type yang:counter32;
  description
  "The number of initiations of LSPs received that were responded by the
  local PCEP entity by sending a PCErr message."
}
} //initiation
} //stateful

//stateful path-key {
  when "../../pcep:role = 'pcc'"
  + "or"
  + "../../pcep:role = 'pcc-and-pce'" {
    description
    "Valid for PCEP Peer as PCC"
  } if-feature "pcep:path-key";
  description
  "If Path-Key is supported";
  leaf num-unknown-path-key {
    type yang:counter32;
    description
    "The number of attempts to expand an unknown path-key."
  }
  leaf num-exp-path-key {
    type yang:counter32;
description
"The number of attempts to expand an expired path-key.";
}
leaf num-dup-path-key {
  type yang:counter32;
  description  
    "The number of duplicate attempts to expand same path-key.";
}
leaf num-path-key-no-attempt {
  type yang:counter32;
  description  
    "The number of expired path-keys with no attempt to expand it.";
}
}
//path-key

//pcep-stats
/*
  * Augment modules to add statistics
  */

augment "/pcep:pcep/pcep:entity/pcep:peers/pcep:peer" {
  description  
    "Augmenting the statistics";
  leaf num-sess-setup-ok {
    type yang:counter32;
    config false;
    description  
      "The number of PCEP sessions successfully established with the peer, including any current session. This counter is incremented each time a session with this peer is successfully established.";
  }
  leaf num-sess-setup-fail {
    type yang:counter32;
    config false;
    description  
      "The number of PCEP sessions with the peer that have been attempted but failed before being fully established. This counter is incremented each time a session retry to this peer fails.";
  }
  container pcep-stats {
  }

config false;
description
"The container for all statistics at peer level.";
uses pcep-stats {
  description
  "Since PCEP sessions can be ephemeral, the peer statistics
  tracks a peer even when no PCEP session currently exists
to that peer. The statistics contained are an aggregate of
the statistics for all successive sessions to that peer.";
}
leaf num-req-sent-closed {
  when "../../pcep:role = 'pce'"
    + "or "
    + "../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCE";
}
type yang:counter32;
description
"The number of requests that were sent to the peer and
implicitly cancelled when the session they were sent
over was closed.";
}
leaf num-req-rcvd-closed {
  when "../../../pcep:role = 'pcc'"
    + "or "
    + "../../../pcep:role = 'pcc-and-pce'"
    description
    "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
"The number of requests that were received from the peer
and implicitly cancelled when the session they were
received over was closed.";
}

//pcep-stats

//augment

augment "/pcep:pcep/pcep:entity/pcep:peers/pcep:peer/"
  + "pcep:sessions/pcep:session" {
  description
  "Augmenting the statistics";
  container pcep-stats {
    description
    "";
  }
}
11. Security Considerations

The YANG modules defined in this document is designed to be accessed via network management protocol such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a pre-configured subset of all available NETCONF or RESTCONF protocol operations and content.

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

- /pcep/entity/ - configure local parameters, capabilities etc.
- /pcep/entity/peers - configure remote peers to setup PCEP session.

Unauthorized access to above list can adversely affect the PCEP session between the local entity and the peers. This may lead to inability to compute new paths, stateful operations on the delegated as well as PCE-initiated LSPs.

Some of the readable data nodes in the ietf-pcep 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:

/pcep/lsp-db - All the LSPs in the network. Unauthorized access to this could provide the all path and network usage information.

/pcep/path-keys/ - The Confidential Path Segments (CPS) are hidden using path-keys. Unauthorized access to this could leak confidential path information.

Some of the RPC operations in the ietf-pcep 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:

trigger-resync - trigger resynchronization with the PCE. Unauthorized access to this could force a PCEP session into continuous state synchronization.

Further, this document also include another YANG module (called ietf-pcep-stats) for maintaining the statistics by augmenting the ietf-pcep YANG module. There are no data nodes defined in this module which are writable/creatable/deletable (i.e., config true). The readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. The statistics could provide information related to the current usage patterns of the network. It is thus important to control read access (e.g., via get, get-config, or notification).

12. IANA Considerations

This document request the IANA to register two URIs in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registrations are requested -

Registrant Contact: The IESG
XML:        N/A; the requested URI is an XML namespace.

Registrant Contact: The IESG
XML:        N/A; the requested URI is an XML namespace.

This document request the IANA to register two YANG modules in the "YANG Module Names" registry [RFC6020], as follows -

13. Implementation Status

[Note to the RFC Editor - remove this section before publication, as well as remove the reference to RFC 7942.]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

Currently, there are no known implementations of the YANG Module as specified.

14. Acknowledgements

The initial document is based on the PCEP MIB [RFC7420]. We would like to thank the authors of aforementioned documents.

Thanks to Mahesh Jethmalani for the Yang Doctor review.

Thanks to Martin Bjorklund and Tom Petch for providing guidance regarding the Yang guidelines.
15. References

15.1. Normative References


15.2. Informative References


Appendix A. Example

The example below provide an overview of PCEP peer session informations and LSP-DB in the Yang Module.

```
+-------+                  +-------+                  +-------+
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCC1  |<----------------> |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
+-------+                  |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCE   |                  |       |                  |       |
IP:192.0.2.1                |       |                  |       |
|       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCC2  |<----------------> |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
+-------+                  |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCE   |                  |       |                  |       |
IP:192.0.2.2                |       |                  |       |
|       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCC4  |<----------------> |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
+-------+                  |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCE   |                  |       |                  |       |
IP:2001:DB8::4              |       |                  |       |
|       |
|       |                  |       |
|       |                  |       |                  |       |
| PCC2  |                  |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
+-------+                  |       |                  |       |
|       |                  |       |                  |       |
|       |                  |       |                  |       |
| PCE   |                  |       |                  |       |
IP:192.0.2.3                |       |                  |       |
2001:DB8::3
|       |
|       |
|       |
|       |

at PCE:
{
  "entity": [
  {
    "addr": "192.0.2.3",
    "oper-status": "oper-status-up",
    "role": "pce",
    "capability": {
      "stateful": {
        "enabled": true
      }
    }
  },
  "lsp-db": [
    "lsp": {
      "plsp-id": 3,
      "plsp-dest": "192.0.2.1",
      "plsp-flags": "tunnel",
      "plsp-label": 10
    }
  ]
}
"pcc-id": "192.0.2.1",
"source": "192.0.2.1",
"destination": "192.0.2.4"
"tunnel-id": 16,
"lsp-id": 3,
"extended-tunnel-id": 0,
"oper-status": "oper-status-up",
"delegated": true,
"symbolic-path-name": "iewauh",
},
"lsp": {
  "plsp-id": 4,
  "pcc-id": "192.0.2.2",
  "source": "192.0.2.2",
  "destination": "192.0.2.5"
  "tunnel-id": 17,
  "lsp-id": 4
  "extended-tunnel-id": 0,
  "oper-status": "oper-status-up",
  "delegated": true,
  "symbolic-path-name": "iewauhiewauh",
  "extended-tunnel-id": 0
}

"peers": [
  {
    "peer": {
      "addr": "192.0.2.1",
      "role": "pcc",
      "capability": {
        "stateful": {
          "enabled": true,
          "active": yes,
        }
      }
    }
  },
  {
    "peer": {
      "addr": "192.0.2.1",
      "role": "pcc",
      "capability": {
        "stateful": {
          "enabled": true,
          "active": yes,
        }
      }
    }
  }
]
"addr": "192.0.2.2",
"role": "pcc",

"capability": {
  "stateful": {
    "enabled": true,
    "active": true,
  }
}

"sessions": [ {
  "session": {
    "initiator": "remote",
    "role": "pcc",
  }
}
}

}

{
  "addr": "2001:DB8::3",
  "oper-status": "oper-status-up",
  "role": "pce",
  "peers": [ {
    "peer": {
      "addr": "2001:DB8::4",
      "role": "pcc",
      "sessions": [ {
        "session": {
          "initiator": "remote",
          "role": "pcc",
        }
      }
    }
  }
}
Appendix B. Contributor Addresses

Rohit Pobbathi
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
EMail: rohit.pobbathi@huawei.com

Vinod KumarS
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India
EMail: vinods.kumar@huawei.com

Zafar Ali
Cisco Systems
Canada
EMail: zali@cisco.com

Xufeng Liu
Volta Networks
EMail: xufeng.liu.ietf@gmail.com

Young Lee
Futurewei
5340 Legacy Drive, Building 3
Plano, TX 75023
USA
EMail: younglee.tx@gmail.com

Udayasree Palle
EMail: udayasreereddy@gmail.com

Xian Zhang
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R.China
EMail: zhang.xian@huawei.com
Avantika
ECI Telecom
India

EMail: avantika.srm@gmail.com

Shashikanth
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: shashikanth.vh@huawei.com

Authors’ Addresses

Dhruv Dhody (editor)
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

EMail: dhruv.ietf@gmail.com

Jonathan Hardwick
Metaswitch
100 Church Street
Enfield EN2 6BQ
UK

EMail: jonathan.hardwick@metaswitch.com

Vishnu Pavan Beeram
Juniper Networks
USA

EMail: vbeeram@juniper.net

Jeff Tantsura
USA

EMail: jefftant@gmail.com

A YANG Data Model for Resource Reservation Protocol (RSVP)
draft-ietf-teas-yang-rsvp-11

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 5, 2020.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
1. Introduction

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

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

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

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

1.2. Model Tree Diagram

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

1.3. Prefixes in Data Node Names

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

```
+-----------+--------------------+-----------+
<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>rt-type</td>
<td>ietf-routing-types</td>
<td>XX</td>
</tr>
<tr>
<td>key-chain</td>
<td>ietf-key-chain</td>
<td>XX</td>
</tr>
</tbody>
</table>
+-----------+--------------------+-----------+
```

Table 1: Prefixes and corresponding YANG modules

2. Model Overview

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

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

2.1. Module(s) Relationship

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

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

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

```
+--------------+
| Routing      |
| ietf-routing |
| +------------+
| o:          |
| +-----------+
| RSVP module |
| ietf-rsvp   |
| +-----------+
| o:          |
| +-----------+
| RSVP extended module |
| ietf-rsvp-extended |
| +-------------------+
```

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

2.2. Design Considerations

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

- Authentication ([RFC2747])
- Refresh Reduction ([RFC2961])
- Hellos ([RFC3209])
- Graceful Restart ([RFC3473], [RFC5063])
The extended RSVP YANG model covers the configuration for optional features that are not must for basic RSVP protocol operation.

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

2.3. Model Notifications

Notifications data modeling is key in any defined data model. [I-D.ietf-netconf-subscribed-notifications] and [I-D.ietf-netconf-yang-push] define a subscription and push mechanism for YANG datastores. This mechanism currently allows the user to:

- Subscribe notifications on a per client basis
- Specify subtree filters or xpath filters so that only interested contents will be sent.
- Specify either periodic or on-demand notifications.

2.4. RSVP Base YANG Model

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

The derived state data is contained in "read-only" nodes directly under the intended object as shown in Figure 2.
module: ietf-rsvp
   -- rw rsvp!
   -- rw globals
     .
   -- rw interfaces
     .
     -- ro <<derived state associated with interfaces>>
     .
   -- rw neighbors
     .
     -- ro <<derived state associated with the tunnel>>
     .
   -- rw sessions
     .
     -- ro <<derived state associated with the tunnel>>

rpcs:
   -- x clear-session
   -- x clear-neighbor

Figure 2: RSVP high-level tree model view

Configuration and state data are grouped to those applicable on per node (global), per interface, per neighbor, or per session.

Global Data:

    The global data cover the configuration and state that is applicable the RSVP protocol behavior.

Interface Data:

    The interface data configuration and state model relevant attributes applicable to one or all RSVP interfaces. Any data or state at the "interfaces" container level is equally applicable to all interfaces – unless overridden by explicit configuration or state under a specific interface.

Neighbor Data:

    The neighbor data cover configuration and state relevant to RSVP neighbors. Neighbors can be dynamically discovered using RSVP signaling or explicitly configured.
Session Data:

The sessions data branch covers configuration and state relevant to RSVP sessions. This is usually derived state that is result of signaling. This model defines attributes related to IP RSVP sessions as defined in [RFC2205].

2.4.1. Tree Diagram

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

```yang
module: ietf-rsvp
augment /rt:routing/rt:control-plane-protocols
/r/rt:control-plane-protocol:
  +--rw rsvp!
    ++--rw globals
    |  ++--rw session-ip* [destination protocol-id destination-port]
    |      |  ++--ro destination-port inet:port-number
    |      |  ++--ro protocol-id uint8
    |      |  ++--ro source? inet:ip-address
    |      |  ++--ro destination inet:ip-address
    |      |  ++--ro session-name? string
    |      |  ++--ro session-state? enumeration
    |      |  ++--ro session-type? identityref
    |      |  ++--ro psbs
    |      |     |  ++--ro psb* []
    |      |     |      |  ++--ro source-port? inet:port-number
    |      |     |      |  ++--ro expires-in? uint32
    |      |  ++--ro rsbs
    |      |     |  ++--ro rsb* []
    |      |     |      |  ++--ro source-port? inet:port-number
    |      |     |      |  ++--ro reservation-style? identityref
    |      |     |      |  ++--ro expires-in? uint32
    +--ro statistics
      |  ++--ro messages
      |      |  ++--ro ack-sent? yang:counter64
      |      |  ++--ro ack-received? yang:counter64
      |      |  ++--ro bundle-sent? yang:counter64
      |      |  ++--ro bundle-received? yang:counter64
      |      |  ++--ro hello-sent? yang:counter64
      |      |  ++--ro hello-received? yang:counter64
      |      |  ++--ro integrity-challenge-sent? yang:counter64
      |      |  ++--ro integrity-challenge-received? yang:counter64
      |      |  ++--ro integrity-response-sent? yang:counter64
      |      |  ++--ro integrity-response-received? yang:counter64
```

++--ro notify-sent? yang:counter64
++--ro notify-received? yang:counter64
++--ro path-sent? yang:counter64
++--ro path-received? yang:counter64
++--ro path-err-sent? yang:counter64
++--ro path-err-received? yang:counter64
++--ro path-tear-sent? yang:counter64
++--ro path-tear-received? yang:counter64
++--ro resv-sent? yang:counter64
++--ro resv-received? yang:counter64
++--ro resv-confirm-sent? yang:counter64
++--ro resv-confirm-received? yang:counter64
++--ro resv-err-sent? yang:counter64
++--ro resv-err-received? yang:counter64
++--ro resv-tear-sent? yang:counter64
++--ro resv-tear-received? yang:counter64
++--ro summary-refresh-sent? yang:counter64
++--ro summary-refresh-received? yang:counter64
++--ro unknown-messages-received? yang:counter64
++--ro packets
| ++--ro sent? yang:counter64
| ++--ro received? yang:counter64
++--ro errors
| ++--ro authenticate? yang:counter64
| ++--ro checksum? yang:counter64
| ++--ro packet-length? yang:counter64
++--rw graceful-restart
| ++--rw enabled? boolean
++--rw interfaces
| ++--rw refresh-reduction
| | ++--rw enabled? boolean
| ++--rw hellos
| | ++--rw enabled? boolean
| ++--rw authentication
| | ++--rw enabled? boolean
| | ++--rw authentication-key? string
| | ++--rw crypto-algorithm identityref
++--ro statistics
| ++--ro messages
| | ++--ro ack-sent? identityref
| | ++--ro ack-received? identityref
| | ++--ro bundle-sent? identityref
| | ++--ro bundle-received? identityref
| | ++--ro hello-sent? identityref
| | ++--ro hello-received? identityref
| | ++--ro integrity-challenge-sent? identityref
| | ++--ro integrity-challenge-received? identityref
| | ++--ro integrity-response-sent? identityref
++--ro integrity-response-received? yang:counter64
++--ro notify-sent? yang:counter64
++--ro notify-received? yang:counter64
++--ro path-sent? yang:counter64
++--ro path-received? yang:counter64
++--ro path-err-sent? yang:counter64
++--ro path-err-received? yang:counter64
++--ro path-tear-sent? yang:counter64
++--ro path-tear-received? yang:counter64
++--ro resv-sent? yang:counter64
++--ro resv-received? yang:counter64
++--ro resv-confirm-sent? yang:counter64
++--ro resv-confirm-received? yang:counter64
++--ro resv-err-sent? yang:counter64
++--ro resv-err-received? yang:counter64
++--ro resv-tear-sent? yang:counter64
++--ro resv-tear-received? yang:counter64
++--ro summary-refresh-sent? yang:counter64
++--ro summary-refresh-received? yang:counter64
++--ro unknown-messages-received? yang:counter64
++--ro packets
  | ++--ro sent? yang:counter64
  | ++--ro received? yang:counter64
++--ro errors
  | ++--ro authenticate? yang:counter64
  | ++--ro checksum? yang:counter64
  | ++--ro packet-length? yang:counter64
++--rw interface* [interface]
  ++--rw interface if:interface-ref
  | ++--rw refresh-reduction
  |  | ++--rw enabled? boolean
  | ++--rw hellos
  |  | ++--rw enabled? boolean
  | ++--rw authentication
  |  | ++--rw enabled? boolean
  |  | ++--rw authentication-key? string
  |  | ++--rw crypto-algorithm identityref
++--ro statistics
  | ++--ro messages
  |  | ++--ro ack-sent? yang:counter64
  |  | ++--ro ack-received? yang:counter64
  |  | ++--ro bundle-sent? yang:counter64
  |  | ++--ro bundle-received? yang:counter64
  |  | ++--ro hello-sent? yang:counter64
  |  | ++--ro hello-received? yang:counter64
  |  | ++--ro integrity-challenge-sent? yang:counter64
  |  | ++--ro integrity-challenge-received? yang:counter64
  |  | ++--ro integrity-response-sent? yang:counter64
  |  | ++--ro integrity-response-received? yang:counter64
++--ro integrity-response-received?  yang:counter64
++--ro notify-sent?  yang:counter64
++--ro notify-received?  yang:counter64
++--ro path-sent?  yang:counter64
++--ro path-received?  yang:counter64
++--ro path-err-sent?  yang:counter64
++--ro path-err-received?  yang:counter64
++--ro path-tear-sent?  yang:counter64
++--ro path-tear-received?  yang:counter64
++--ro resv-sent?  yang:counter64
++--ro resv-received?  yang:counter64
++--ro resv-confirm-sent?  yang:counter64
++--ro resv-confirm-received?  yang:counter64
++--ro resv-err-sent?  yang:counter64
++--ro resv-err-received?  yang:counter64
++--ro resv-tear-sent?  yang:counter64
++--ro resv-tear-received?  yang:counter64
++--ro summary-refresh-sent?  yang:counter64
++--ro summary-refresh-received?  yang:counter64
++--ro unknown-messages-received?  yang:counter64
++--ro packets
|++--ro sent?  yang:counter64
|++--ro received?  yang:counter64
++--ro errors
++--ro authenticate?  yang:counter64
++--ro checksum?  yang:counter64
++--ro packet-length?  yang:counter64
++--rw neighbors
++--rw neighbor* [address]
|++--rw address  inet:ip-address
|++--rw epoch?  uint32
|++--rw expiry-time?  uint32
++--rw graceful-restart
|++--rw enabled?  boolean
|++--rw local-restart-time?  uint32
|++--rw local-recovery-time?  uint32
|++--rw neighbor-restart-time?  uint32
|++--rw neighbor-recovery-time?  uint32
|++--rw helper-mode
||++--rw enabled?  boolean
||++--rw max-helper-restart-time?  uint32
||++--rw max-helper-recovery-time?  uint32
||++--rw neighbor-restart-time-remaining?  uint32
||++--rw neighbor-recovery-time-remaining?  uint32
++--rw hello-status?  enumeration
++--rw interface?  if:interface-ref
++--rw neighbor-state?  enumeration
++--rw refresh-reduction-capable?  boolean
2.4.2. YANG Module

The ietf-rsvp module imports from the following modules:

- ietf-interfaces defined in [RFC8343]
- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-routing defined in [RFC8349]
- ietf-key-chain defined in [RFC8177]

<CODE BEGINS> file "ietf-rsvp@2019-07-04.yang"
module ietf-rsvp {
  yang-version 1.1;

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

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

Figure 3: RSVP model tree diagram
import ietf-interfaces {
    prefix if;
    reference "RFC8343: A YANG Data Model for Interface Management";
}
import ietf-inet-types {
    prefix inet;
    reference "RFC6991: Common YANG Data Types";
}
import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
}
import ietf-routing {
    prefix "rt";
    reference "RFC8349: A YANG Data Model for Routing Management (NMDA Version)";
}
import ietf-key-chain {
    prefix "key-chain";
    reference "RFC8177: YANG Data Model for Key Chains";
}
organization
    "IETF Traffic Engineering Architecture and Signaling (TEAS) Working Group";
contact
    "WG Web: <http://tools.ietf.org/wg/teas/>"
    "WG List: <mailto:teas@ietf.org>"
    "Editor: Vishnu Pavan Beeram <mailto: vbeeram@juniper.net>"
    "Editor: Tarek Saad <mailto: tsaad@juniper.net>"
    "Editor: Rakesh Gandhi <mailto: rgandhi@cisco.com>"
    "Editor: Xufeng Liu <mailto: xufeng.liu.ietf@gmail.com>"
    "Editor: Igor Bryskin"
description
"This module contains the RSVP YANG data model.
The model fully conforms to the Network Management Datastore Architecture (NMDA).

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

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

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

revision "2019-07-04" {
  description
    "A YANG Data Model for Resource Reservation Protocol";
  reference
    "RFCXXXX: A YANG Data Model for Resource Reservation Protocol (RSVP)";
}

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

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

identity rsvp-session-ip {
  base rsvp-session-type;
  description "RSVP IP session type";
}
identity reservation-style {
    description "Base identity for reservation style";
}

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

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

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

grouping graceful-restart-config {
    description "Base configuration parameters relating to RSVP Graceful-Restart";
    leaf enabled {
        type boolean;
        description "'true' if RSVP Graceful Restart is enabled. 'false' if RSVP Graceful Restart is disabled.";
    }
}

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

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

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

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

grouping authentication-config {
  description
    "Configuration parameters relating to RSVP authentication";
  leaf enabled {
    type boolean;
    description
      "'true' if RSVP Authentication is enabled.
      'false' if RSVP Authentication is disabled."
  }
  leaf authentication-key {
    type string;
    description
      "An authentication key string";
    reference
      "RFC 2747: RSVP Cryptographic Authentication";
  }
  leaf crypto-algorithm {
    type identityref {
      base key-chain:crypto-algorithm;
    }
    mandatory true;
    description
      "Cryptographic algorithm associated with key."
  }
}
grouping authentication {
    description "Top level grouping for RSVP authentication parameters";
    container authentication {
        description "Top level container for RSVP authentication parameters";
        uses authentication-config;
    }
}

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

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

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

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

grouping session-attributes-state {

description
  "Top level grouping for RSVP session properties";
leaf destination-port {
  type inet:port-number;
  description "RSVP destination port";
  reference "RFC2205";
}
leaf protocol-id {
  type uint8;
  description "The IP protocol ID.";
  reference "RFC2205, section 3.2";
}
leaf source {
  type inet:ip-address;
  description "RSVP source address";
  reference "RFC2205";
}
leaf destination {
  type inet:ip-address;
  description "RSVP destination address";
  reference "RFC2205";
}
leaf session-name {
  type string;
  description
    "The signaled name of this RSVP session.";
}
leaf session-state {
  type enumeration {
    enum "up" {
      description
        "RSVP session is up";
    }
    enum "down" {
      description
        "RSVP session is down";
    }
  }
  description
    "Enumeration of RSVP session states";
}
leaf session-type {
  type identityref {
    base rsvp-session-type;
  }
  description "RSVP session type";
}
container psbs {
description "Path State Block container";
list psb {
    description "List of path state blocks";
    leaf source-port {
        type inet:port-number;
        description "RSVP source port";
        reference "RFC2205";
    }
    leaf expires-in {
        type uint32;
        units seconds;
        description "Time to reservation expiry (in seconds)";
    }
}
}
}
}
}

grouping neighbor-attributes {
    description "Top level grouping for RSVP neighbor properties";
    leaf address {
        type inet:ip-address;
        description "Address of RSVP neighbor";
    }
}

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

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

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

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

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

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

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

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

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

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

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

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

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

  leaf neighbor-recovery-time-remaining {
    type uint32;
    units seconds;
    description
    "Number of seconds remaining for neighbor to
     refresh.";
  }
} // helper-mode
} // graceful-restart
leaf hello-status {
  type enumeration {
    enum "enabled" {
      description
      "Enabled";
    }
    enum "disabled" {
      description
      "Disabled";
    }
    enum "restarting" {
      description
      "Restarting";
    }
  }
  description
  "Hello status";
}

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

leaf neighbor-state {
  type enumeration {
    enum "up" {
      description
      "up";
    }
    enum "down" {
      description
      "down";
    }
    enum "hello-disable" {
      description
      "hello-disable";
    }
    enum "restarting" {
      description
      "restarting";
    }
  }
  description
  "Neighbor state";
}
leaf refresh-reduction-capable {
    type boolean;
    description
        "enables all RSVP refresh reduction message
         bundling, RSVP message ID, reliable message delivery
         and summary refresh";
    reference
        "RFC 2961 RSVP Refresh Overhead Reduction
         Extensions";
}

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

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

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

grouping protocol-state {
    description
        "RSVP protocol statistics grouping";
    container messages {

description

"RSVP protocol statistics container";
leaf ack-sent {
  type yang:counter64;
  description
  "Hello sent count";
}
leaf ack-received {
  type yang:counter64;
  description
  "Hello received count";
}
leaf bundle-sent {
  type yang:counter64;
  description
  "Bundle sent count";
}
leaf bundle-received {
  type yang:counter64;
  description
  "Bundle received count";
}
leaf hello-sent {
  type yang:counter64;
  description
  "Hello sent count";
}
leaf hello-received {
  type yang:counter64;
  description
  "Hello received count";
}
leaf integrity-challenge-sent {
  type yang:counter64;
  description
  "Integrity Challenge sent count";
}
leaf integrity-challenge-received {
  type yang:counter64;
  description
  "Integrity Challenge received count";
leaf integrity-response-sent {
  type yang:counter64;
  description
    "Integrity Response sent count";
}

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

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

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

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

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

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

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

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

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

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

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

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

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

leaf resv-err-received {
    type yang:counter64;
    description
        "Resv error received count";
grouping errors-state {
  description "Error statistics state grouping";
  container errors {
    description "Error statistics state container";
    leaf authenticate {
      type yang:counter64;
      description "The total number of packets received with an authentication failure.";
    }
  }
  leaf checksum {

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

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

grouping statistics-state {
  description "RSVP statistic attributes.";
  container statistics {
    config false;
    description
      "statistics state container";
    uses protocol-state;
    uses packets-state;
    uses errors-state;
  }
}

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

grouping global-attributes {
  description
    "Top level grouping for RSVP global properties";
  container sessions {
    description
      "RSVP sessions container";
    list session-ip {
      key "destination protocol-id destination-port";
      config false;
      description
        "List of RSVP sessions";
      uses session-attributes-state;
    }
    uses session-attributes-state;
  }
}
uses statistics-state;

)

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

augment "/rt:routing/rt:control-plane-protocols/"  
+ "rt:control-plane-protocol" {  
when "rt:type = 'rsvp:rsvp'" {  
description  
"This augment is only valid when routing protocol  
instance type is RSVP.";
}

description  
"RSVP protocol augmentation";
container rsvp {  
presence "Enable RSVP feature";
description "RSVP feature container";
container globals {  
description "RSVP global properties.";
uses global-attributes;
uses graceful-restart;
  }

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

list interface {  
key "interface";

description  
"RSVP interfaces.";
leaf interface {  
type if:interface-ref;

description  
"RSVP interface.";
  }
uses intf-attributes;
  }
  }
container neighbors {
    description "RSVP neighbors container";
    list neighbor {
        key "address";
        description "List of RSVP neighbors";
        uses neighbor-attributes;
    }
}

grouping session-ref {
    description "Session reference information";
    leaf destination {
        type leafref {
        }
        mandatory true;
        description "RSVP session";
    }
    leaf protocol-id {
        type uint8;
        mandatory true;
        description "The RSVP session protocol ID";
    }
    leaf destination-port {
        type inet:ip-address;
        mandatory true;
        description "The RSVP session destination port";
    }
}
rpc clear-session {
    description "Clears RSVP sessions RPC";
    input {
        leaf routing-protocol-instance-name {
            type leafref {
                path "/rt:routing/rt:control-plane-protocols/" + "rt:control-plane-protocol/rt:name";
            }
            mandatory "true";
            description "Name of the RSVP protocol instance whose session is being cleared."
        }
        mandatory "true";
        description "If the corresponding RSVP instance doesn’t exist,"
then the operation will fail with an error-tag of 'data-missing' and an error-app-tag of 'routing-protocol-instance-not-found'.

choice filter-type {
  mandatory true;
  description "Filter choice";
  case match-all {
    leaf all {
      type empty;
      mandatory true;
      description "Match all RSVP sessions";
    }
  }
  case match-one {
    container session-info {
      description "Specifies the specific session to invoke operation on";
      choice session-type {
        mandatory true;
        description "RSVP session type";
        case rsvp-session-ip {
          uses session-ref;
        }
      }
    }
  }
}

rpc clear-neighbor {
  description "RPC to clear the RSVP Hello session to a neighbor";
  input {
    leaf routing-protocol-instance-name {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/" + "rt:control-plane-protocol/rt:name";
      }
      mandatory "true";
      description "Name of the RSVP protocol instance whose session is being cleared."
    }
  }
  if the corresponding RSVP instance doesn’t exist, then the operation will fail with an error-tag of 'data-missing' and an error-app-tag of
2.5. RSVP Extended YANG Model

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

2.5.1. Tree Diagram

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

```yaml
module: ietf-rsvp-extended
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
      /rsvp:graceful-restart:
        rw restart-time? uint32
        rw recovery-time? uint32
    augment /rt:routing/rt:control-plane-protocols
      /rt:control-plane-protocol/rsvp:rsvp/rsvp:globals
```

Internet-Draft            RSVP YANG Data Model                 July 2019

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

Figure 4: RSVP extended model tree diagram

2.5.2. YANG Module

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

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

Figure 5 shows the RSVP extended YANG module:

<CODE BEGINS> file "ietf-rsvp-extended@2019-07-04.yang"
module ietf-rsvp-extended {
  yang-version 1.1;


  prefix "rsvp-ext";


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

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

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

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

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

contact
  "WG Web:  <http://tools.ietf.org/wg/teas/>
  WG List:  <mailto:teas@ietf.org>
  Editor:  Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
  Editor:  Tarek Saad
            <mailto:tsaad@juniper.net>
  Editor:  Rakesh Gandhi
            <mailto:rgandhi@cisco.com>
  Editor:  Himanshu Shah
            <mailto:hshah@ciena.com>
  Editor:  Xufeng Liu
            <mailto:Xufeng_Liu@jabil.com>
  Editor:  Xia Chen
description
"This module contains the Extended RSVP YANG data model. The model fully conforms to the Network Management Datastore Architecture (NMDA).

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

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

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices."

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

revision "2019-07-04" {
  description
    "A YANG Data Model for Extended Resource Reservation Protocol";
  reference
    "RFCXXXX: A YANG Data Model for Resource Reservation Protocol (RSVP)";
}

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

feature error-statistics {
  description

"Indicates support for error statistics";
}

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

feature graceful-restart {
  description
    "Indicates support for RSVP graceful restart";
}

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

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

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

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

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

grouping graceful-restart-extended-config {
  description
    "Configuration parameters relating to RSVP Graceful-Restart";
  leaf restart-time {
    type uint32;
    units seconds;
    description
      "Graceful restart time (seconds).";
    reference
      Beeram, et al.           Expires January 5, 2020               
      [Page 36]
leaf recovery-time {
  type uint32;
  units seconds;
  description "RSVP state recovery time";
}

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

range "1..10000";
}
description
  "Number of retransmits when messages are
dropped.";
reference
  "RFC 2747: RSVP Cryptographic
  Authentication";
}
leaf key-chain {
  type key-chain:key-chain-ref;
  description
    "Key chain name to authenticate RSVP
    signaling messages.";
  reference
    "RFC 2747: RSVP Cryptographic
    Authentication";
}
}
grouping hellos-extended-config {
  description
    "Configuration parameters relating to RSVP
    hellos";
  leaf interface-based {
    type empty;
    description
      "Enable interface-based Hello adjacency if present.";
  }
  leaf hello-interval {
    type uint32;
    units milliseconds;
    description
      "Configure interval between successive Hello
      messages in milliseconds.";
    reference
      "RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels.
      RFC 5495: Description of the Resource
      Reservation Protocol - Traffic-Engineered
      (RSVP-TE) Graceful Restart Procedures";
  }
  leaf hello-misses {
    type uint32 {
      range "1..10";
    }
    description
      "Configure max number of consecutive missed
      Hello messages.";
  }
reference
}
}
grouping signaling-parameters-extended-config {
  description
    "Configuration parameters relating to RSVP signaling";
  leaf refresh-interval {
    type uint32;
    description
      "Set interval between successive refreshes";
  }
  leaf refresh-misses {
    type uint32;
    description
      "Set max number of consecutive missed messages for state expiry";
  }
  leaf checksum {
    type boolean;
    description
      "Enable RSVP message checksum computation";
  }
  leaf patherr-state-removal {
    type empty;
    description
      "State-Removal flag in Path Error message if present.";
  }
}

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

  leaf bundle-message-max-size {
    type uint32 {
      range "512..65000";
    }
  }
  description
    "Configure maximum size (bytes) of a
single RSVP Bundle message.

leaf reliable-ack-hold-time {
  type uint32;
  units milliseconds;
  description
    "Configure hold time in milliseconds for
    sending RSVP ACK message(s).";
}

leaf reliable-ack-max-size {
  type uint32;
  description
    "Configure max size of a single RSVP ACK
    message.";
}

leaf reliable-retransmit-time {
  type uint32;
  units milliseconds;
  description
    "Configure min delay in milliseconds to
    wait for an ACK before a retransmit.";
}

leaf reliable-srefresh {
  type empty;
  description
    "Configure use of reliable messaging for
    summary refresh if present.";
}

leaf summary-max-size {
  type uint32 {
    range "20..65000";
  }
  description
    "Configure max size (bytes) of a single
    RSVP summary refresh message.";
}

}

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

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

leaf in-dropped {
    type yang:counter64;
    description
        "In packet drop count";
}

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

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

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

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

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

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

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

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

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

/**
 * RSVP all interfaces extensions
 */

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

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

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

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

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

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

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

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

Figure 5: RSVP extended YANG module

3. IANA Considerations

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

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

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

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

name:       ietf-rsvp
prefix:     ietf-rsvp
reference:  RFCXXXX

name:       ietf-rsvp-extended
prefix:     ietf-rsvp-extended
reference:  RFCXXXX
4. Security Considerations

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

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

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

/routing/routing-control-plane-protocols/routing-control-plane-protocol/rsvp:

The presence of this container enables the RSVP protocol functionality on a device. It also controls the configuration settings on data nodes pertaining to RSVP sessions, interfaces and neighbors. All of which are considered sensitive and if access to either of these is compromised, it can result in temporary network outages or be employed to mount DoS attacks.

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

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

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

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

6. Contributors

Himanshu Shah
Ciena

Email: hshah@ciena.com

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones
Brocade

Email: raqib@Brocade.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

7. Normative References

[I-D.ietf-netconf-subscribed-notifications]

[I-D.ietf-netconf-yang-push]


Authors’ Addresses

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Tarek Saad
Juniper Networks

Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems, Inc.

Email: rgandhi@cisco.com

Xufeng Liu
Jabil

Email: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei Technologies

Email: Igor.Bryskin@huawei.com
A YANG Data Model for Traffic Engineering Tunnels and Interfaces
draft-ietf-teas-yang-te-21

Abstract

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

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

Status of This Memo

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

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

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

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

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

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

Table of Contents

1. Introduction .................................................. 3
   1.1. Terminology ........................................... 3
   1.2. Prefixes in Data Node Names .......................... 3
   1.3. TE Technology Models .................................. 4
   1.4. State Data Organization ............................... 4
2. Model Overview ................................................ 4
   2.1. Module(s) Relationship ................................ 5
   2.2. Design Considerations ................................ 7
   2.3. Model Tree Diagram .................................... 7
3. Model Organization ............................................ 48
   3.1. Global Configuration and State Data .................. 49
   3.2. Interfaces Configuration and State Data ............... 50
   3.3. Tunnels Configuration and State Data .................. 51
       3.3.1. Tunnel Compute-Only Mode ......................... 51
       3.3.2. Tunnel Hierarchical Link Endpoint ................. 52
   3.4. TE LSPs State Data ..................................... 52
   3.5. Global RPC Data ....................................... 52
   3.6. Interface RPC Data .................................... 52
   3.7. Tunnel RPC Data ....................................... 52
4. TE Generic and Helper YANG Modules ........................ 53
5. IANA Considerations ............................................ 99
6. Security Considerations ....................................... 99
7. Acknowledgement ............................................... 100
8. Contributors ................................................ 100
9. References .................................................... 101
   9.1. Normative References ................................. 101
   9.2. Informative References ............................... 104
Authors’ Addresses ............................................ 104
1. Introduction

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

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

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

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

1.1. Terminology

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

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

1.2. Prefixes in Data Node Names

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

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

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

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

1.4. State Data Organization

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

2. Model Overview

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

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

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

The TE data model for specific instances of signaling protocol are outside the scope of this document and are defined in other documents. For example, the RSVP-TE YANG model augmentation of the TE model is covered in [I-D.ietf-teas-yang-rsvp].
Figure 1: Relationship of TE module(s) with other signaling protocol modules

Figure 2: Relationship between generic and technology specific TE types modules
2.2. Design Considerations

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

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

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

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

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

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

2.3. Model Tree Diagram

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

module: ietf-te
  +--rw te!
    +--rw globals
      +--rw named-admin-groups
        +--rw named-admin-group* [name]
          +--rw name string
          +--rw bit-position? uint32
        +--rw named-srlgs
          +--rw named-srlg* [name] {te-types:named-srlg-groups}?
            +--rw name string
            +--rw group? te-types:srlg
            +--rw cost? uint32
          +--rw named-path-constraints
            +--rw named-path-constraint* [name]
              {te-types:named-path-constraints}?
                +--rw name string
+++rw te-bandwidth
  +++rw (technology)?
    ++++: (generic)
      +++rw generic? te-bandwidth
  +++rw link-protection? identityref
  +++rw setup-priority? uint8
  +++rw hold-priority? uint8
  +++rw signaling-type? identityref
++rw path-metric-bounds
  +++rw path-metric-bound* [metric-type]
    +++rw metric-type identityref
    +++rw upper-bound? uint64
++rw path-affinities-values
  +++rw path-affinities-value* [usage]
    +++rw usage identityref
    +++rw value? admin-groups
++rw path-affinity-names
  +++rw path-affinity-name* [usage]
    +++rw usage identityref
    +++rw affinity-name* [name]
      +++rw name string
++rw path-srlgs-lists
  +++rw path-srlgs-list* [usage]
    +++rw usage identityref
    +++rw values* srlg
++rw path-srlgs-names
  +++rw path-srlgs-name* [usage]
    +++rw usage identityref
    +++rw names* string
++rw disjointness?
  te-path-disjointness
++rw explicit-route-objects-always
  +++rw route-object-exclude-always* [index]
    +++rw index uint32
    ++++: (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 srlg
  +++-rw srlg?  uint32
++-rw shared-resources-tunnels
  +++-rw lsp-shared-resources-tunnel*  tunnel-ref
++-rw path-in-segment!
  +++-rw label-restrictions
    +++-rw label-restriction* [index]
      +++-rw restriction?  enumeration
      +++-rw index  uint32
      +++-rw label-start
        +++-rw te-label
        +++-rw (technology)?
          -: (generic)
          +++-rw generic?
            rt-types:generalized-label
          +++-rw direction?  te-label-direction
        +++-rw label-end
      +++-rw te-label
        +++-rw (technology)?
          -: (generic)
        +++-rw generic?
          rt-types:generalized-label
        +++-rw direction?  te-label-direction
    +++-rw label-step
      +++-rw (technology)?
        -: (generic)
        +++-rw generic?  int32
      +++-rw range-bitmap?  yang:hex-string
++-rw path-out-segment!
  +++-rw label-restrictions
    +++-rw label-restriction* [index]
      +++-rw restriction?  enumeration
      +++-rw index  uint32
      +++-rw label-start
        +++-rw te-label
        +++-rw (technology)?
          -: (generic)
          +++-rw generic?
            rt-types:generalized-label
          +++-rw direction?  te-label-direction
        +++-rw label-end
      +++-rw te-label
        +++-rw (technology)?
          -: (generic)
        +++-rw generic?
          rt-types:generalized-label
        +++-rw direction?  te-label-direction
      +++-rw label-step
++-rw (technology)?
  +--rw generic? int32
  +--rw range-bitmap? yang:hex-string
++-rw te-dev:lsp-install-interval? uint32
++-rw te-dev:lsp-cleanup-interval? uint32
++-rw te-dev:lsp-invalidation-interval? uint32
+-rw tunnels
  ++-rw tunnel* [name]
    +--ro operational-state? identityref
    +-rw name string
    +-rw identifier? uint16
    +-rw description? string
    +-rw encoding? identityref
    +-rw switching-type? identityref
    +-rw provisioning-state? identityref
    +-rw preference? uint8
    +-rw reoptimize-timer? uint16
    +--rw source?
      |  te-types:te-node-id
    +--rw destination?
      |  te-types:te-node-id
    +--rw src-tp-id? yang:hex-string
    +--rw dst-tp-id? yang:hex-string
    +--rw bidirectional? boolean
  +--rw association-objects
    ++-rw association-object* [type ID source global-source]
      +--rw type identityref
      +--rw ID uint16
      +--rw source te-types:te-node-id
      +--rw global-source te-types:te-node-id
    ++-rw association-object-extended* [type ID source global-source extended-ID]
      +--rw type identityref
      +--rw ID uint16
      +--rw source te-types:te-node-id
      +--rw global-source te-types:te-node-id
      +--rw extended-ID yang:hex-string
  +--rw protection
    ++-rw enable? boolean
    ++-rw protection-type? identityref
    ++-rw protection-reversion-disable? boolean
    ++-rw hold-off-time? uint32
    ++-rw wait-to-revert? uint16
    ++-rw aps-signal-id? uint8
  +--rw restoration
    ++-rw enable? boolean
    ++-rw restoration-type? identityref
++rw restoration-scheme? identityref
++rw restoration-reversion-disable? boolean
++rw hold-off-time? uint32
++rw wait-to-restore? uint16
++rw wait-to-revert? uint16
++rw te-topology-identifier
  ++rw provider-id? te-global-id
  ++rw client-id? te-global-id
  ++rw topology-id? te-topology-id
++rw te-bandwidth
  ++rw (technology)?
  +--:(generic)
    ++rw generic? te-bandwidth
++rw link-protection? identityref
++rw setup-priority? uint8
++rw hold-priority? uint8
++rw signaling-type? identityref
++rw dependency-tunnels
  ++rw dependency-tunnel* [name]
    ++rw name
    |   -> ../../../tunnels/tunnel/name
    ++rw encoding? identityref
    ++rw switching-type? identityref
++rw hierarchical-link
  ++rw local-te-node-id? te-types:te-node-id
  ++rw local-te-link-tp-id? te-types:te-tp-id
  ++rw remote-te-node-id? te-types:te-node-id
++rw te-topology-identifier
  ++rw provider-id? te-global-id
  ++rw client-id? te-global-id
  ++rw topology-id? te-topology-id
++rw p2p-primary-paths
  ++rw p2p-primary-path* [name]
    ++rw name
    ++rw path-setup-protocol? identityref
    ++rw path-computation-method? identityref
    ++rw path-computation-server?
      inet:ip-address
    ++rw compute-only? empty
    ++rw use-path-computation? boolean
    ++rw lockdown? empty
    ++ro path-scope? identityref
    ++rw optimizations
    +--rw (algorithm)?
      +--:(metric) (path-optimization-metric)?
        ++rw optimization-metric* [metric-type]
          +--rw metric-type identityref
| ---rw tiebreaker* [tiebreaker-type] |
|   +--rw tiebreaker-type  identityref |
|   +:(objective-function) |
|     +--:(path-optimization-objective-function)? |
|       +--rw objective-function |
|       +--rw objective-function-type? |
|         identityref |
| ---rw preference?  uint8 |
| ---rw k-requested-paths?  uint8 |
| ---rw named-path-constraint?  leafref |
|   +: {te-types:named-path-constraints}? |
| ---rw te-bandwidth |
|   +--rw (technology)? |
|     +:(generic) |
|     +--rw generic?  te-bandwidth |
|   +--rw link-protection?  identityref |
|   +--rw setup-priority?  uint8 |
|   +--rw hold-priority?  uint8 |
| ---rw signaling-type?  identityref |
| ---rw path-metric-bounds |
|   +--rw path-metric-bound* [metric-type] |
|     +--rw metric-type  identityref |
|     +--rw upper-bound?  uint64 |
| ---rw path-affinities-values |
|   +--rw path-affinities-value* [usage] |
|     +--rw usage  identityref |
|     +--rw value?  admin-groups |
| ---rw path-affinity-names |
|   +--rw path-affinity-name* [usage] |
|     +--rw usage  identityref |
|     +--rw affinity-name* [name] |
|     +--rw name  string |
| ---rw path-srlgs-lists |
|   +--rw path-srlgs-list* [usage] |
|     +--rw usage  identityref |
|     +--rw values*  srlg |
| ---rw path-srlgs-names |
|   +--rw path-srlgs-name* [usage] |
|     +--rw usage  identityref |
|     +--rw names*  string |
| ---rw disjointness? |
|   te-path-disjointness |
| ---rw explicit-route-objects-always |
|   +--rw route-object-exclude-always* [index] |
|     +--rw index  uint32 |
|     +--rw (type)? |
|     +:(numbered-node-hop) |
|     +--rw numbered-node-hop |
+--rw (technology)?
   ++--:(generic)
       ++--rw generic?
           rt-types:generalized-label
   ++--rw direction?
       te-label-direction

++--rw label-end
   ++--rw te-label
       ++--rw (technology)?
           ++--:(generic)
               ++--rw generic?
                   rt-types:generalized-label
               ++--rw direction?
                   te-label-direction

++--rw label-step
   ++--rw (technology)?
       ++--:(generic)
           ++--rw generic?  int32
   ++--rw range-bitmap?  yang:hex-string

++--ro computed-paths-properties
   ++--ro computed-path-properties*  [k-index]
       ++--ro k-index  uint8
   ++--ro path-properties
      ++--ro path-metric*  [metric-type]
         ++--ro metric-type  identityref
      ++--ro accumulative-value?  uint64
   ++--ro path-affinities-values
      ++--ro path-affinities-value*  [usage]
         ++--ro usage  identityref
      ++--ro value?  admin-groups

++--ro path-affinity-names
   ++--ro path-affinity-name*  [usage]
      ++--ro usage  identityref
   ++--ro affinity-name*  [name]
      ++--ro name  string

++--ro path-srlgs-lists
   ++--ro path-srlgs-list*  [usage]
      ++--ro usage  identityref
   ++--ro values*  srlg

++--ro path-srlgs-names
   ++--ro path-srlgs-name*  [usage]
      ++--ro usage  identityref
   ++--ro names*  string

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

---ro path-route-objects

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

++-ro te-dev:upstream-info
  ++-ro te-dev:nhop? inet:ip-address
  ++-ro te-dev:neighbor? inet:ip-address
  ++-ro te-dev:label?
    | rt-types:generalized-label

++-rw p2p-primary-reverse-path
  ++-rw name? string
  ++-rw path-setup-protocol? identityref
  ++-rw path-computation-method? identityref
  ++-rw path-computation-server?
    | inet:ip-address
  ++-rw compute-only? empty
  ++-rw use-path-computation? boolean
  ++-rw lockdown? empty
  ++-ro path-scope? identityref
  ++-rw optimizations
    | ++-rw (algorithm)?
      | ++-(metric) (path-optimization-metric)?
        | ++-rw optimization-metric* [metric-type]
          | ++-rw metric-type
            | identityref
          | ++-rw weight?
            | uint8
        | ++-rw explicit-route-exclude-objects
          | ++-rw route-object-exclude-object* [index]
            | ++-rw index
              | uint32
          | ++-rw (type)?
            | ++-(numbered-node-hop)
              | ++-rw numbered-node-hop
                | ++-rw node-id
                  | te-node-id
                | ++-rw hop-type?
                  | te-hop-type
          | ++-(numbered-link-hop)
            | ++-rw numbered-link-hop
              | ++-rw link-tp-id
                | te-tp-id
```plaintext
++-:(label)
  +--rw label-hop
    ++-rw te-label
    ++-rw (technology)?
      +--:(generic)
        +--rw generic?
        rt-types:generalized-label
    ++-rw direction?
      te-label-direction
  +-rw route-object-include-exclude* [index]
    +--rw explicit-route-usage?
      | identityref
    +--rw index uint32
  +--rw (type)?
    +--:(numbered-node-hop)
      +--rw numbered-node-hop
      ++-rw node-id te-node-id
      ++-rw hop-type? te-hop-type
    +--:(numbered-link-hop)
      +--rw numbered-link-hop
      ++-rw link-tp-id te-tp-id
      ++-rw hop-type? te-hop-type
      ++-rw direction?
        te-link-direction
    +--:(unnumbered-link-hop)
      +--rw unnumbered-link-hop
      ++-rw link-tp-id te-tp-id
      ++-rw node-id te-node-id
      ++-rw hop-type? te-hop-type
      ++-rw direction?
        te-link-direction
    +--:(as-number)
      +--rw as-number-hop
      ++-rw as-number inet:as-number
      ++-rw hop-type? te-hop-type
    +--:(label)
      +--rw label-hop
      ++-rw te-label
      ++-rw (technology)?
        +--:(generic)
        +--rw generic?
        rt-types:generalized-label
      ++-rw direction?
        te-label-direction
    +--:(srlg)
      +--rw srlg
      ++-rw srlg? uint32
      ++-rw shared-resources-tunnels
```

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

---ro node-id            te-node-id
---ro flags*             path-attribute-flags
---:(numbered-link-hop)
  ---ro numbered-link-hop
     ---ro link-tp-id    te-tp-id
     ---ro flags*       path-attribute-flags
---:(unnumbered-link-hop)
  ---ro unnumbered-link-hop
     ---ro link-tp-id    te-tp-id
     ---ro node-id?     te-node-id
     ---ro flags*       path-attribute-flags
---:(label)
  ---ro label-hop
   ---ro te-label
    ---ro (technology)?
     | +---:(generic)
     |     ---ro generic?
     | rt-types:generalized-label
    ---ro direction?
             te-label-direction
     ---ro flags*       path-attribute-flags
  ---ro path-properties
   ---ro path-metric* [metric-type]
    ---ro metric-type
       | identityref
    ---ro accumulative-value? uint64
  ---ro path-affinities-values
   ---ro path-affinities-value* [usage]
    ---ro usage identityref
    ---ro value? admin-groups
  ---ro path-affinity-names
   ---ro path-affinity-name* [usage]
    ---ro usage identityref
    ---ro affinity-name* [name]
     ---ro name string
  ---ro path-srlgs-lists
   ---ro path-srlgs-list* [usage]
    ---ro usage identityref
    ---ro values* srlg
  ---ro path-srlgs-names
   ---ro path-srlgs-name* [usage]
    ---ro usage identityref
    ---ro names* string
++-ro path-route-objects
++-ro path-computed-route-object*
  [index]
    ++-ro index
      |       uint32
    ++-ro (type)?
      ++-(numbered-node-hop)
        ++-ro numbered-node-hop
          ++-ro node-id
            |     te-node-id
          ++-ro hop-type?
            te-hop-type
      ++-(numbered-link-hop)
        ++-ro numbered-link-hop
          ++-ro link-tp-id
            |     te-tp-id
          ++-ro hop-type?
            te-hop-type
          ++-ro direction?
            te-link-direction
      ++-(unnumbered-link-hop)
        ++-ro unnumbered-link-hop
          ++-ro link-tp-id
            |     te-tp-id
          ++-ro node-id
            |     te-node-id
          ++-ro hop-type?
            te-hop-type
          ++-ro direction?
            te-link-direction
      ++-(as-number)
        ++-ro as-number-hop
          ++-ro as-number
            |     inet:as-number
          ++-ro hop-type?
            te-hop-type
      ++-(label)
        ++-ro label-hop
          ++-ro te-label
            ++-ro (technology)?
              ++-(generic)
                ++-ro generic?
                  rt-types:generalized-label
            ++-ro direction?
              te-label-direction
        ++-ro shared-resources-tunnels
          ++-ro lsp-shared-resources-tunnel*
            tunnel-ref
++ rw p2p-secondary-reverse-path
   +-- rw secondary-path? leafref
   +-- rw path-setup-protocol? identityref
++ rw candidate-p2p-secondary-paths
   +-- rw candidate-p2p-secondary-path* [secondary-path]
      +-- rw secondary-path leafref
      +-- rw path-setup-protocol? identityref
      +-- ro active? boolean
++ rw p2p-secondary-paths
   +-- rw p2p-secondary-path* [name]
      +-- rw name string
      +-- rw path-setup-protocol? identityref
      +-- rw path-computation-method? identityref
      +-- rw path-computation-server? \[inet:ip-address\]
      +-- ro compute-only? empty
      +-- ro use-path-computation? boolean
      +-- ro lockdown? empty
      +-- ro path-scope? identityref
   ++ rw optimizations
      +-- rw (algorithm)?
         +--:(metric) [path-optimization-metric]?
            +-- rw optimization-metric* [metric-type]
               +-- rw metric-type identityref
               +-- ro 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? \[te-hop-type\]
                        +-- ro direction? \[te-link-direction\]
++-rw upper-bound?        uint64
+-rw path-affinities-values
  ++-rw path-affinities-value* [usage]
    ++-rw usage     identityref
    ++-rw value?    admin-groups
+-rw path-affinity-names
  ++-rw path-affinity-name* [usage]
    ++-rw usage     identityref
    ++-rw affinity-name* [name]
      ++-rw name      string
+-rw path-srlgs-lists
  ++-rw path-srlgs-list* [usage]
    ++-rw usage     identityref
    ++-rw values*   srlg
+-rw path-srlgs-names
  ++-rw path-srlgs-name* [usage]
    ++-rw usage     identityref
    ++-rw names*   string
+-rw disjointness?
  te-path-disjointness
+-rw explicit-route-objects-always
  ++-rw route-object-exclude-always* [index]
    ++-rw index        uint32
    ++-rw (type)?
      ++-:(numbered-node-hop)
        ++-rw numbered-node-hop
          ++-rw node-id     te-node-id
          ++-rw hop-type?   te-hop-type
      ++-:(numbered-link-hop)
        ++-rw numbered-link-hop
          ++-rw link-tp-id    te-tp-id
          ++-rw hop-type?   te-hop-type
          ++-rw direction?  te-link-direction
      ++-:(unnumbered-link-hop)
        ++-rw unnumbered-link-hop
          ++-rw link-tp-id    te-tp-id
          ++-rw node-id       te-node-id
          ++-rw hop-type?    te-hop-type
          ++-rw direction?  te-link-direction
      ++-:(as-number)
        ++-rw as-number-hop
          ++-rw as-number    inet:as-number
          ++-rw hop-type?    te-hop-type
      ++-:(label)
        ++-rw label-hop
          ++-rw te-label
            ++-rw (technology)?
              ++-:(generic)
++-rw generic?
    rt-types:generalized-label
++-rw direction?
    te-label-direction
+-rw route-object-include-exclude* [index]
+-rw explicit-route-usage? identityref
++-rw index uint32
++-rw (type)?
    ++-(numbered-node-hop)
      ++-rw numbered-node-hop
        ++-rw node-id te-node-id
        ++-rw hop-type? te-hop-type
    ++-(numbered-link-hop)
      ++-rw numbered-link-hop
        ++-rw link-tp-id te-tp-id
        ++-rw hop-type? te-hop-type
        ++-rw direction? te-link-direction
    ++-(unnumbered-link-hop)
      ++-rw unnumbered-link-hop
        ++-rw link-tp-id te-tp-id
        ++-rw node-id te-node-id
        ++-rw hop-type? te-hop-type
        ++-rw direction? te-link-direction
    ++-(as-number)
      ++-rw as-number-hop
        ++-rw as-number inet:as-number
        ++-rw hop-type? te-hop-type
    ++-(label)
      ++-rw label-hop
        ++-rw te-label
          ++-rw (technology)?
            ++-(generic)
              ++-rw generic?
                rt-types:generalized-label
            ++-rw direction?
              te-label-direction
        ++-(srlg)
          ++-rw srlg
            ++-rw srlg? uint32
      ++-rw shared-resources-tunnels
        ++-rw lsp-shared-resources-tunnel* tunnel-ref
      ++-rw path-in-segment!
        ++-rw label-restrictions
          ++-rw label-restriction* [index]
            ++-rw restriction? enumeration
            ++-rw index uint32
          ++-rw label-start
            ++-rw te-label
---rw (technology)?
  +--:(generic)
  |  ---rw generic?
  |  |  rt-types:generalized-label
  |  ---rw direction?
  |  te-label-direction
---rw label-end
  ---rw te-label
  ---rw (technology)?
  |  +--:(generic)
  |  |  ---rw generic?
  |  |  rt-types:generalized-label
  |  |  ---rw direction?
  |  |  te-label-direction
  ---rw label-step
  ---rw (technology)?
  |  +--:(generic)
  |  |  ---rw generic?  int32
  |  |  ---rw range-bitmap?  yang:hex-string
  ---rw path-out-segment!
  ---rw label-restrictions
  ---rw label-restriction* [index]
  ---rw restriction?  enumeration
  ---rw index  uint32
  ---rw label-start
  ---rw te-label
  ---rw (technology)?
  |  +--:(generic)
  |  |  ---rw generic?
  |  |  rt-types:generalized-label
  |  |  ---rw direction?
  |  |  te-label-direction
  ---rw label-end
  ---rw te-label
  ---rw (technology)?
  |  +--:(generic)
  |  |  ---rw generic?
  |  |  rt-types:generalized-label
  |  |  ---rw direction?
  |  |  te-label-direction
  ---rw label-step
  ---rw (technology)?
  |  +--:(generic)
  |  |  ---rw generic?  int32
  |  |  ---rw range-bitmap?  yang:hex-string
  ---rw protection
  |  ---rw enable?  boolean
  |  ---rw protection-type?  identityref
++-rw protection-reversion-disable?   boolean
++-rw hold-off-time?               uint32
++-rw wait-to-revert?             uint16
++-rw aps-signal-id?              uint8

++-rw restoration
   ++-rw enable?                 boolean
   ++-rw restoration-type?      identityref
   ++-rw restoration-scheme?    identityref
   ++-rw restoration-reversion-disable?   boolean
   ++-rw hold-off-time?         uint32
   ++-rw wait-to-restore?       uint16
   ++-rw wait-to-revert?        uint16

++-ro computed-paths-properties
   ++-ro computed-path-properties* [k-index]
      ++-ro k-index             uint8
      ++-ro path-properties
         ++-ro path-metric* [metric-type]
            ++-ro metric-type   identityref
            ++-ro accumulative-value?   uint64
         ++-ro path-affinities-values
            ++-ro path-affinities-value* [usage]
               ++-ro usage         identityref
               ++-ro value?      admin-groups
         ++-ro path-affinity-names
            ++-ro path-affinity-name* [usage]
               ++-ro usage         identityref
               ++-ro affinity-name* [name]
                  ++-ro name      string
         ++-ro path-srlgs-lists
            ++-ro path-srlgs-list* [usage]
               ++-ro usage         identityref
               ++-ro values*    srlg
         ++-ro path-srlgs-names
            ++-ro path-srlgs-name* [usage]
               ++-ro usage         identityref
               ++-ro names*    string
         ++-ro path-route-objects
            ++-ro path-computed-route-object* [index]
               ++-ro index       uint32
               ++-ro (type)?
                  +++:(numbered-node-hop)
                     ++-ro numbered-node-hop
                        ++-ro node-id    te-node-id
                        ++-ro hop-type?  te-hop-type
                  +++:(numbered-link-hop)
                     ++-ro numbered-link-hop
                        ++-ro link-tp-id  te-tp-id
+--ro hop-type?
  |  te-hop-type
++--ro direction?
  +--ro link-type?
++--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?
  +--ro link-direction
++--: (as-number)
  +--ro as-number-hop
    +--ro as-number
    inet:as-number
    +--ro hop-type?
    te-hop-type
++--: (label)
  +--ro label-hop
    +--ro te-label
      +--ro (technology)?
        +--: (generic)
          +--ro generic?
            rt-types:generalized-label
          +--ro direction?
            te-label-direction
          +--ro shared-resources-tunnels
            +--ro lsp-shared-resources-tunnel*
              tunnel-ref
++--ro computed-path-error-infos
  +--ro computed-path-error-info* []
    +--ro error-description?  string
    +--ro error-timestamp?  yang:date-and-time
    +--ro error-reason?  identityref
++--ro lsp-provisioning-error-infos
  +--ro lsp-provisioning-error-info* []
    +--ro error-description?  string
    +--ro error-timestamp?  yang:date-and-time
    +--ro error-node-id?  te-types:te-node-id
    +--ro error-link-id?  te-types:te-tp-id
    +--ro lsp-id?  uint16
++--ro lsps
  +--ro lsp* [lsp-id]
    +--ro lsp-provisioning-error-infos
      +--ro lsp-provisioning-error-info* []
        +--ro error-description?  string

++-ro error-timestamp?
    | yang:date-and-time
++-ro error-node-id?
    | te-types:te-node-id
++-ro error-link-id?
    | te-types:te-tp-id
++-ro source?
    | te-types:te-node-id
++-ro destination?
    | te-types:te-node-id
++-ro tunnel-id?
    | uint16
++-ro lsp-id
    | uint16
++-ro extended-tunnel-id?
    | yang:dotted-quad
++-ro operational-state?
    | identityref
++-ro path-setup-protocol?
    | identityref
++-ro origin-type?
    | enumeration
++-ro lsp-resource-status?
    | enumeration
++-ro lockout-of-normal?
    | boolean
++-ro freeze?
    | boolean
++-ro lsp-protection-role?
    | enumeration
++-ro lsp-protection-state?
    | identityref
++-ro protection-group-ingress-node-id?
    | te-types:te-node-id
++-ro protection-group-egress-node-id?
    | te-types:te-node-id
++-ro lsp-shared-resources-tunnel?
    | tunnel-ref
++-ro lsp-record-route-information
    +--ro lsp-record-route-information* [index]
        +--ro index uint32
        +--ro (type)?
            +--:(numbered-node-hop)
                +--ro numbered-node-hop
                    +--ro node-id te-node-id
                    +--ro flags*
                        path-attribute-flags
            +--:(numbered-link-hop)
++-ro numbered-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
    |  (technology)?
    |  +++-:(generic)
    |      ++-ro generic?
    |          rt-types:generalized-label
    |          ++-ro direction?
    |                  te-label-direction
++-ro shared-resources-tunnels
    ++-ro lsp-shared-resources-tunnel*
        tunnel-ref
++-ro te-dev:lsp-timers
    ++-ro te-dev:life-time?  uint32
    ++-ro te-dev:time-to-install?  uint32
    ++-ro te-dev:time-to-destroy?  uint32
++-ro te-dev:downstream-info
    ++-ro te-dev:nhop?
        inet:ip-address
    |  ++-ro te-dev:outgoing-interface?
    |      if:interface-ref
    |  ++-ro te-dev:neighbor?
    |      inet:ip-address
---x tunnel-action
  +--w input
|  ++-w action-type? identityref
  +ro output
  +ro action-result? identityref
---x protection-external-commands
  +w input
  |  +w protection-external-command? identityref
  |  +w protection-group-ingress-node-id?
|  te-types:te-node-id
  |  +w protection-group-egress-node-id?
|  te-types:te-node-id
  |  +w path-ref? path-ref
  |  +w traffic-type? enumeration
  |  +w extra-traffic-tunnel-ref? tunnel-ref
 ---rw te-dev:lsp-install-interval? uint32
 ---rw te-dev:lsp-cleanup-interval? uint32
 ---rw te-dev:lsp-invalidation-interval? uint32
 ---rw tunnel-p2mp* [name]
  +rw name string
  +rw identifier? uint16
  +rw description? string
  +ro operational-state? identityref
---ro lsps-state
  +ro lsp* [source destination tunnel-id lsp-id extended-tunnel-id]
  +ro source
  |  te-types:te-node-id
  +ro destination
  |  te-types:te-node-id
  +ro tunnel-id uint16
  +ro lsp-id uint16
  +ro extended-tunnel-id yang:dotted-quad
  +ro operational-state? identityref
  +ro path-setup-protocol? identityref
  +ro origin-type? enumeration
  +ro lsp-resource-status? enumeration
  +ro lockout-of-normal? boolean
  +ro freeze? boolean
  +ro lsp-protection-role? enumeration
  +ro lsp-protection-state? identityref
++-ro protection-group-ingress-node-id?
    te-types:te-node-id
++-ro protection-group-egress-node-id?
    te-types:te-node-id
++-ro lsp-record-route-information
    ++-ro lsp-record-route-information* [index]
    ++-ro index                        uint32
    ++-ro (type)?
        ++-:(numbered-node-hop)
            ++-ro numbered-node-hop
                ++-ro node-id    te-node-id
                ++-ro flags*     path-attribute-flags
        ++-:(numbered-link-hop)
            ++-ro numbered-link-hop
                ++-ro link-tp-id    te-tp-id
                ++-ro flags*        path-attribute-flags
        ++-:(unnumbered-link-hop)
            ++-ro unnumbered-link-hop
                ++-ro link-tp-id    te-tp-id
                ++-ro node-id?      te-node-id
                ++-ro flags*        path-attribute-flags
        ++-:(label)
            ++-ro label-hop
                ++-ro te-label
                    ++-ro (technology)?
                    ++-:(generic)
                        ++-ro generic?
                            rt-types:generalized-label
                    ++-ro direction?    te-label-direction
                    ++-ro flags*        path-attribute-flags
            ++-ro te-dev:lsp-timers
                ++-ro te-dev:life-time?         uint32
                ++-ro te-dev:time-to-install?   uint32
                ++-ro te-dev:time-to-destroy?   uint32
            ++-ro te-dev:downstream-info
                ++-ro te-dev:nhop?
                    inet:ip-address
                ++-ro te-dev:outgoing-interface?  if:interface-ref
                ++-ro te-dev:neighbor?
                    inet:ip-address
                ++-ro te-dev:label?
                    rt-types:generalized-label
            ++-ro te-dev:upstream-info
                ++-ro te-dev:pnhop?
                    inet:ip-address
                ++-ro te-dev:neighbor?
                    inet:ip-address
                ++-ro te-dev:label?
                    rt-types:generalized-label
            ++-rw te-dev:interfaces
                ++-rw te-dev:threshold-type?        enumeration
                ++-rw te-dev:delta-percentage?    rt-types:percentage
                ++-rw te-dev:threshold-specification?  enumeration
++-rw te-dev:up-thresholds*              rt-types:percentage
++-rw te-dev:down-thresholds*            rt-types:percentage
++-rw te-dev:up-down-thresholds*         rt-types:percentage
++-rw te-dev:interface* [interface]
  ++-rw te-dev:interface
     | if:interface-ref
  ++-rw te-dev:te-metric?
     if:te-types:te-metric
  ++-rw (te-dev:admin-group-type)?
     ++-:(te-dev:value-admin-groups)
        ++-rw (te-dev:value-admin-group-type)?
           ++-:(te-dev:admin-groups)
               ++-rw te-dev:admin-group?
                  if:te-types:admin-group
           ++-:(te-dev:extended-admin-groups)
               {te-types:extended-admin-groups}?
               ++-rw te-dev:extended-admin-group?
                  if:te-types:extended-admin-group
           ++-:(te-dev:named-admin-groups)
               ++-rw te-dev:named-admin-groups* [named-admin-group]
                  ++-rw te-dev:named-admin-group leafref
        ++-:(te-dev:srlg-type)?
           ++-:(te-dev:value-srlgs)
              ++-rw te-dev:values* [value]
                 ++-rw te-dev:value uint32
           ++-:(te-dev:named-srlgs)
              ++-rw te-dev:named-srlgs* [named-srlg]
                 {te-types:named-srlg-groups}?
                 ++-rw te-dev:named-srlg leafref
        ++-rw te-dev:threshold-type?
           enumeration
        ++-rw te-dev:delta-percentage?
           if:rt-types:percentage
        ++-rw te-dev:threshold-specification?
           enumeration
        ++-rw te-dev:up-thresholds*             rt-types:percentage
        ++-rw te-dev:down-thresholds*            rt-types:percentage
        ++-rw te-dev:up-down-thresholds*         rt-types:percentage
        ++-rw te-dev:switching-capabilities* [switching-capability]
           ++-rw te-dev:switching-capability identityref
           ++-rw te-dev:encoding?                   identityref
        ++-ro te-dev:state
           ++-ro te-dev:flood-interval? uint32
           ++-ro te-dev:last-flooded-time? uint32
3. Model Organization

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

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

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

Figure 3: TE generic model configuration and state tree
module: ietf-te
  +--rw te!
    +--rw globals
      .
    .
  +--rw tunnels
    .
  +-- lsps-state

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

Figure 4: TE generic highlevel model view

3.1. Global Configuration and State Data

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

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

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

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

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

Figure 5: TE interface state

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

- Bandwidth information: maximum bandwidth, available bandwidth at different priorities and for each class-type (CT)

- List of admitted LSPs
  - Name, bandwidth value and pool, time, priority

- Statistics: state counters, flooding counters, admission counters (accepted/rejected), preemption counters

- Adjacency information
  - Neighbor address
  - Metric value
3.3. Tunnels Configuration and State Data

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

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

Figure 6: TE interface state tree

Examples of tunnel configuration data for TE tunnels:

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

3.3.1. Tunnel Compute-Only Mode

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

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

3.3.2. Tunnel Hierarchical Link Endpoint

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

3.4. TE LSPs State Data

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

3.5. Global RPC Data

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

- Clear global TE statistics of various features

3.6. Interface RPC Data

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

- Clear TE statistics for all or for individual TE interfaces
- Trigger immediate flooding for one or all TE interfaces

3.7. Tunnel RPC Data

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

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

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

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

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

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

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

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

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

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

  contact
    "WG Web: <http://tools.ietf.org/wg/teas/>
    WG List: <mailto:teas@ietf.org>
    WG Chair: Lou Berger
    <mailto:lberger@labn.net>
description
"YANG data module for TE configuration,
state, RPC and notifications."
}

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

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

identity path-computation-error-no-topology {
  base path-computation-error-reason;
  description
  "Path computation error no topology error reason";
}

identity path-computation-error-no-server {
  base path-computation-error-reason;
  description
  "Path computation error no server error reason";
}

identity path-computation-error-path-not-found {
  base path-computation-error-reason;
}
typedef tunnel-ref {
  type leafref {
    path "te:te/te:tunnels/te:tunnel/te:name";
  }
  description "This type is used by data models that need to reference configured TE tunnel."
}

typedef tunnel-p2mp-ref {
  type leafref {
    path "te:te/te:tunnels/te:tunnel-p2mp/te:name";
  }
  description "This type is used by data models that need to reference configured P2MP TE tunnel.";
  reference "RFC4875";
}

typedef path-ref {
  type union {
    type leafref {
      path "te:te/te:tunnels/te:tunnel/" + "te:p2p-primary-paths/te:p2p-primary-path/te:name";
    }
    type leafref {
      path "te:te/te:tunnels/te:tunnel/" + "te:p2p-secondary-paths/te:p2p-secondary-path/te:name";
    }
  }
  description "This type is used by data models that need to reference configured primary or secondary path of a TE tunnel."
}

/**
 * TE tunnel generic groupings
 */
grouping p2p-secondary-path-properties {
  description "tunnel path properties.";
  uses p2p-path-properties;
  uses path-constraints-common;
  uses protection-restoration-properties;
  uses p2p-path-properties-state;
}
grouping p2p-primary-path-properties {
    description "TE tunnel primary path properties grouping";
    uses p2p-path-properties;
    uses path-constraints-common;
    uses p2p-path-properties-state;
}

grouping path-properties {
    description "TE computed path properties grouping";
    container path-properties {
        description "The TE path computed properties";
        list path-metric {
            key metric-type;
            description "TE path metric type";
            leaf metric-type {
                type identityref {
                    base te-types:path-metric-type;
                }
                description "TE path metric type";
            }
            leaf accumulative-value {
                type uint64;
                description "TE path metric accumulative value";
            }
        }
    }
    uses te-types:generic-path-affinities;
    uses te-types:generic-path-srlgs;
    container path-route-objects {
        config 'false';
        description "Container for the list of computed route objects as
defined by the computation engine";
        list path-computed-route-object {
            key index;
            ordered-by user;
            description "List of computed route objects returned by the
computation engine";
            leaf index {
                type uint32;
                description "Route object entry index. The index is used to
develop an entry in the list. The order of entries is defined by the user without relying on key values";
            }
        }
    }
}
uses te-types:explicit-route-hop;
}
}
uses shared-resources-tunnels;
}
}

grouping p2p-path-properties-state {
  description "TE per path state parameters";
  container computed-paths-properties {
    config 'false';
    description "Computed path properties container";
    list computed-path-properties {
      key k-index;
      description "List of computed paths";
      leaf k-index {
        type uint8;
        description "The k-th path returned from the computation server. A lower k value path is more optimal than higher k value path(s)";
      }
      uses path-properties {
        description "The TE path computed properties";
      }
    }
  }
  uses computed-path-error-info;
  uses lsp-provisioning-error-info {
    augment "lsp-provisioning-error-infos/" + "lsp-provisioning-error-info" {
      description "Augmentation of LSP provisioning information under a specific path";
      leaf lsp-id {
        type uint16;
        description "The LSP-ID for which path computation was performed.";
      }
    }
  }
  container lsps {
    config 'false';
    description "TE LSPs container";
    list lsp {
      key "lsp-id";
      description "List of LSPs associated with the tunnel.";
    }
  }
}
uses lsp-provisioning-error-info;
uses lsp-properties-state;
uses shared-resources-tunnels-state;
uses lsp-record-route-information-state;
uses path-properties {
    description "The TE path actual properties";
}
}
}

grouping computed-path-error-info {
    description "Grouping for path computation error information";
    container computed-path-error-infos {
        config false;
        description "Path computation information container";
        list computed-path-error-info {
            description "List of path computation info entries";
            leaf error-description {
                type string;
                description "Textual representation of the error occurred during path 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 p2p-path-properties-common {
    description
"TE tunnel common path properties configuration grouping";
    leaf name {
        type string;
        description "TE path name";
    }
    leaf path-setup-protocol {
        type identityref {
            base te-types:path-signaling-type;
        }
        default te-types:path-setup-static;
        description
"Signaling protocol used to set up this tunnel";
    }
    leaf path-computation-method {

type identityref {
    base te-types:path-computation-method;
}
default te-types:path-locally-computed;
description
    "The method used for computing the path, either
    locally computed, queried from a server or not
    computed at all (explicitly configured).";
}
leaf path-computation-server {
    when "./path-computation-method = "+ 
    "te-types:path-externally-queried" "{
        description
            "The path-computation server when the path is
            externally queried";
        type inet:ip-address;
        description
            "Address of the external path computation
            server";
    }
}
leaf compute-only {
    type empty;
    description
        "When set, the path is computed and updated whenever
        the topology is updated. No resources are committed
        or reserved in the network.";
}
leaf use-path-computation {
    when "./path-computation-method =" + 
    "te-types:path-locally-computed’";
    type boolean;
    default ‘true’;
    description "A CSPF dynamically computed path";
}
leaf lockdown {
    type empty;
    description
        "Indicates no reoptimization to be attempted for
        this path.";
}
leaf path-scope {
    type identityref {
        base te-types:path-scope-type;
    }
    default te-types:path-scope-end-to-end;
    config ‘false’;
    description "Path scope if segment or an end-to-end path";
grouping p2p-reverse-path-properties {
    description "TE tunnel reverse path properties configuration grouping";
    uses p2p-path-properties-common;
    uses te-types:generic-path-optimization;
    leaf named-path-constraint {
        if-feature te-types:named-path-constraints;
        type leafref {
            path "../../../../../../globals/
            "named-path-constraints/named-path-constraint/
            "name";
        }
        description "Reference to a globally defined named path constraint set";
    }
}

grouping p2p-primary-reverse-path-properties {
    description "TE P2P tunnel primary reverse path properties.";
    reference "RFC7551";
    container p2p-primary-reverse-path {
        description "Tunnel reverse primary path properties";
        uses p2p-reverse-path-properties;
        uses path-constraints-common;
        uses p2p-path-properties-state;
        container p2p-secondary-reverse-path {
            description "Tunnel reverse secondary path properties";
            uses p2p-secondary-reverse-path-properties;
        }
    }
}

grouping p2p-path-properties {
    description "TE tunnel path properties configuration grouping";
    uses p2p-path-properties-common;
    uses te-types:generic-path-optimization;
    leaf preference {
        type uint8 {
            range "1..255";
        }
        default 1;
        description
"Specifies a preference for this path. The lower the number higher the preference";
}
leaf k-requested-paths {
  type uint8;
  default 1;
  description
  "The number of k-shortest-paths requested from the path computation server and returned sorted by its optimization objective";
}
leaf named-path-constraint {
  if-feature te-types:named-path-constraints;
  type leafref {
    path "../../../globals/" + "named-path-constraints/named-path-constraint/" + "name";
  }
  description
  "Reference to a globally defined named path constraint set";
}
}

grouping hierarchical-link-properties {
  description
  "Hierarchical link grouping";
  reference "RFC4206";
  container hierarchical-link {
    description
    "Identifies a hierarchical link (in client layer) that this tunnel is associated with.";
    leaf local-te-node-id {
      type te-types:te-node-id;
      default "0.0.0.0";
      description
      "Local TE node identifier";
    }
    leaf local-te-link-tp-id {
      type te-types:te-tp-id;
      default 0;
      description
      "Local TE link termination point identifier";
    }
    leaf remote-te-node-id {
      type te-types:te-node-id;
      default "0.0.0.0";
      description
    }
  }
}

"Remote TE node identifier";
}
uses te-types:te-topology-identifier;
}

grouping protection-restoration-properties-state {
  description
    "Protection parameters grouping";
  leaf lockout-of-normal {
    type boolean;
    default 'false';
    description
      "When set to 'True', it represents a lockout of normal traffic external command. When set to 'False', it represents a clear lockout of normal traffic external command. The lockout of normal traffic command applies to this Tunnel."
    reference "RFC4427";
  }
  leaf freeze {
    type boolean;
    default 'false';
    description
      "When set to 'True', it represents a freeze external command. When set to 'False', it represents a clear freeze external command. The freeze command 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
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 p2p-dependency-tunnels-properties {
    description
    "Grouping for tunnel dependency list of tunnels";
    container dependency-tunnels {
        description "Dependency tunnels list";
        list dependency-tunnel {
            key "name";
            description "Dependency tunnel entry";
            leaf name {
                type leafref {
                    path "../../../../../tunnels/tunnel/name";
                    require-instance 'false';
                }
                description "Dependency tunnel name";
            }
            leaf encoding {
                type identityref {
                    base te-types:lsp-encoding-types;
                }
                default te-types:lsp-encoding-packet;
                description "LSP encoding type";
                reference "RFC3945";
            }
            leaf switching-type {
                type identityref {
                    base te-types:switching-capabilities;
                }
                default te-types:switching-pscl;
                description "LSP switching type";
                reference "RFC3945";
            }
        }
    }
}

grouping tunnel-p2p-config {
    description
"Configuration parameters relating to TE tunnel";
leaf name {
    type string;
    description "TE tunnel name.";
}
leaf identifier {
    type uint16;
    description
        "TE tunnel Identifier.";
    reference "RFC3209";
}
leaf description {
    type string;
    default 'None';
    description
        "Textual description for this TE tunnel";
}
leaf encoding {
    type identityref {
        base te-types:lsp-encoding-types;
    }
    default te-types:lsp-encoding-packet;
    description "LSP encoding type";
    reference "RFC3945";
}
leaf switching-type {
    type identityref {
        base te-types:switching-capabilities;
    }
    default te-types:switching-pscl;
    description "LSP switching type";
    reference "RFC3945";
}
leaf provisioning-state {
    type identityref {
        base te-types:tunnel-state-type;
    }
    default te-types:tunnel-state-up;
    description "TE tunnel administrative state.";
}
leaf preference {
    type uint8 {
        range "1..255";
    }
    default 100;
    description
        "Specifies a preference for this tunnel. 
        A lower number signifies a better preference";
}
leaf reoptimize-timer {
  type uint16;
  units seconds;
  description
    "frequency of reoptimization of a traffic engineered LSP";
}
leaf source {
  type te-types:te-node-id;
  description "TE tunnel source node ID.";
}
leaf destination {
  type te-types:te-node-id;
  description "TE tunnel destination node ID";
}
leaf src-tp-id {
  type yang:hex-string;
  default '00:00:00:00';
  description
    "TE tunnel source termination point identifier.";
}
leaf dst-tp-id {
  type yang:hex-string;
  default '00:00:00:00';
  description
    "TE tunnel destination termination point identifier.";
}
leaf bidirectional {
  type boolean;
  default 'false';
  description "TE tunnel bidirectional";
}
uses tunnel-p2p-associations-properties;
uses protection-restoration-properties;
uses te-types:tunnel-constraints;
uses p2p-dependency-tunnels-properties;
uses hierarchical-link-properties;

grouping tunnel-p2p-associations-properties {
  description "TE tunnel association grouping";
  container association-objects {
    description "TE tunnel associations";
    list association-object {
      key "type ID source global-source";
      description "List of association base objects";
      reference "RFC4872";
      leaf type {
type identityref {
    base te-types:association-type;
} description "Association type";
reference "RFC4872";

leaf ID {
    type uint16;
    description "Association ID";
    reference "RFC4872";
}

leaf source {
    type te-types:te-node-id;
    description "Association source";
    reference "RFC4872";
}

leaf global-source {
    type te-types:te-node-id;
    description "Association global source";
    reference "RFC4872";
}

list association-object-extended {
    key "type ID source global-source extended-ID";
    description "List of extended association objects";
    reference "RFC6780";
    leaf type {
        type identityref {
            base te-types:association-type;
        } description "Association type";
    }
    leaf ID {
        type uint16;
        description "Association ID";
        reference "RFC4872";
    }
    leaf source {
        type te-types:te-node-id;
        description "Association source";
    }
    leaf global-source {
        type te-types:te-node-id;
        description "Association global source";
        reference "RFC4872";
    }
    leaf extended-ID {
        type yang:hex-string;
    }
}
description "Association extended ID";
    reference "RFC4872";
}
}
}

grouping path-access-segment-info {
    description "If an end-to-end tunnel crosses multiple domains using
the same technology, some additional constraints have to be
taken in consideration in each domain";
    container path-in-segment {
        presence
        "The end-to-end tunnel starts in a previous domain;
this tunnel is a segment in the current domain.";
        description
        "This tunnel is a segment that needs to be coordinated
with previous segment stitched on head-end side.";
        uses te-types:label-set-info;
    }
    container path-out-segment {
        presence
        "The end-to-end tunnel is not terminated in this domain;
this tunnel is a segment in the current domain.";
        description
        "This tunnel is a segment that needs to be coordinated
with previous segment stitched on head-end side.";
        uses te-types:label-set-info;
    }
}

/* TE tunnel configuration/state grouping */
grouping tunnel-p2mp-properties {
    description "Top level grouping for P2MP tunnel properties.";
    leaf name {
        type string;
        description "TE tunnel name.";
    }
    leaf identifier {
        type uint16;
        description "TE tunnel Identifier.";
        reference "RFC3209";
    }
    leaf description {
        type string;
    }
}
default 'None';
description
  "Textual description for this TE tunnel";
}
leaf operational-state {
  type identityref {
    base te-types:tunnel-state-type;
  }
default te-types:tunnel-state-up;
  config 'false';
  description "TE tunnel administrative state.";
}
}
grouping p2p-path-candidate-secondary-path-config {
  description
    "Configuration parameters relating to a secondary path which
    is a candidate for a particular primary path"
leaf secondary-path {
  type leafref {
    path "../../../../../p2p-secondary-paths/" +
        "p2p-secondary-path/name";
  }
  description
    "A reference to the secondary path that should be utilised
    when the containing primary path option is in use";
}
leaf path-setup-protocol {
  type identityref {
    base te-types:path-signaling-type;
  }
  default te-types:path-setup-static;
  description
    "Signaling protocol used to set up this tunnel";
}
}
grouping p2p-secondary-reverse-path-properties {
  description
    "Configuration parameters relating to a secondary path which
    is a candidate for a particular primary path"
leaf secondary-path {
  type leafref {
    path "../../../../../p2p-secondary-paths/" +
        "p2p-secondary-path/name";
  }
  description
    "A reference to the secondary path that should be utilised
    when the containing primary path option is in use";
}
leaf path-setup-protocol {
  type identityref {
    base te-types:path-signaling-type;
  }
  default te-types:path-setup-static;
  description
    "Signaling protocol used to set up this tunnel";
}
}
leaf path-setup-protocol {
    type identityref {
        base te-types:path-signaling-type;
    }
    default te-types:path-setup-static;
    description "Signaling protocol used to set up this tunnel";
}

grouping tunnel-p2p-properties {
    description "Top level grouping for tunnel properties.";
    leaf operational-state {
        type identityref {
            base te-types:tunnel-state-type;
        }
        default te-types:tunnel-state-up;
        config 'false';
        description "TE tunnel administrative state.";
    }
    uses tunnel-p2p-config;
    container p2p-primary-paths {
        description "Set of P2P primary paths container";
        list p2p-primary-path {
            key "name";
            description "List of primary paths for this tunnel.";
            uses p2p-primary-path-properties;
            uses p2p-primary-reverse-path-properties;
            container candidate-p2p-secondary-paths {
                description "The set of candidate secondary paths which may be used for this primary path. When secondary paths are specified in the list the path of the secondary LSP in use must be restricted to those path options referenced. The priority of the secondary paths is specified within the list. Higher priority values are less preferred - that is to say that a path with priority 0 is the most preferred path. In the case that the list is empty, any secondary path option may be utilised when the current primary path is in use.";
            }
        }
    }
}
list candidate-p2p-secondary-path {
  key "secondary-path";
  description
    "List of secondary paths for this tunnel."
  uses p2p-path-candidate-secondary-path-config;

  leaf active {
    type boolean;
    config 'false';
    description
      "Indicates the current active path option that has
       been selected of the candidate secondary paths";
  }
}
}
}
}

container p2p-secondary-paths {
  description "Set of P2P secondary paths container";
  list p2p-secondary-path {
    key "name";
    description
      "List of secondary paths for this tunnel.";
    uses p2p-secondary-path-properties;
  }
}
}
}
grouping shared-resources-tunnels-state {
  description
    "The specific tunnel that is using the shared secondary path
     resources";
  leaf lsp-shared-resources-tunnel {
    type tunnel-ref;
    description
      "Reference to the tunnel that sharing secondary path
       resources with this tunnel";
  }
}
}

grouping shared-resources-tunnels {
  description
    "Set of tunnels that share secondary path resources with
     this tunnel";
  container shared-resources-tunnels {
    description
      "Set of tunnels that share secondary path resources with
       this tunnel";
    leaf-list lsp-shared-resources-tunnel {

type tunnel-ref;
description
  "Reference to the tunnel that sharing secondary path
   resources with this tunnel";
}
}

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

grouping tunnel-protection-actions {
  description
    "Protection external command actions";
  action protection-external-commands {
    input {
      leaf protection-external-command {
        type identityref {
          base te-types:protection-external-commands;
        }
        description
          "Protection external command";
      }
      leaf protection-group-ingress-node-id {
        type te-types:te-node-id;
        description
          "When specified, indicates whether the action is
           applied on ingress node.
           By default, if neither ingress nor egress node-id
is set, the 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."
}
grouping lsp-record-route-information-state {
    description "recorded route information grouping";
    container lsp-record-route-information {
        description "RSVP recorded route object information";
        list lsp-record-route-information {
            when ".paralleled-origin-type = 'ingress'" {
                description "Applicable on ingress LSPs only";
            }
            key "index";
            description "Record route list entry";
            uses te-types:record-route-state;
        }
    }
}

grouping lsps-state-grouping {
    description "LSPs state operational data grouping";
    container lsps-state {
        config 'false';
        description "TE LSPs state container";
        list lsp {
            key
            "source destination tunnel-id lsp-id "+
            "extended-tunnel-id";
            description "List of LSPs associated with the tunnel.";
            uses lsp-properties-state;
            uses lsp-record-route-information-state;
        }
    }
}

/*** End of TE LSP groupings ***/

/**
 * TE global generic groupings
 */

/* Global named admin-groups configuration data */
grouping named-admin-groups-properties {
  description "Global named administrative groups configuration grouping";
  leaf name {
    type string;
    description "A string name that uniquely identifies a TE interface named admin-group";
  }
  leaf bit-position {
    type uint32;
    description "Bit position representing the administrative group";
    reference "RFC3209 and RFC7308";
  }
}

/* Global named admin-srlgs configuration data */
grouping named-srlgs-properties {
  description "Global named SRLGs configuration grouping";
  leaf name {
    type string;
    description "A string name that uniquely identifies a TE interface named srlg";
  }
  leaf group {
    type te-types:srlg;
    description "An SRLG value";
  }
}
leaf cost {
    type uint32;
    description
      "SRLG associated cost. Used during path to append
       the path cost when traversing a link with this SRLG";
} }

grouping named-srlgs {
    description
      "Global named SRLGs configuration grouping";
    container named-srlgs {
        description "TE named SRLGs container";
        list named-srgl {
            if-feature te-types:named-srlg-groups;
            key "name";
            description
              "A list of named SRLG groups";
            uses named-srlgs-properties;
        }
    }
}

/* Global named paths constraints configuration data */
grouping path-constraints-state {
    description "TE path constraints state";
    leaf bandwidth {
        type te-types:te-bandwidth;
        config 'false';
        description
          "A technology agnostic requested bandwidth to use
           for path computation";
    }
    leaf disjointness-type {
        type te-types:te-path-disjointness;
        config 'false';
        description
          "The type of resource disjointness.";
    }
}

grouping path-constraints-common {
    description
      "Global named path constraints configuration grouping";
    uses te-types:common-path-constraints-attributes;
    uses te-types:generic-path-disjointness;
    uses te-types:path-constraints-route-objects;
}
uses shared-resources-tunnels {
    description
    "Set of tunnels that are allowed to share secondary path resources of this tunnel";
}

uses path-access-segment-info {
    description
    "Tunnel constraints induced by other segments."
}

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

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;
    }
}

grouping tunnels-grouping {

description
"Tunnels TE configuration data grouping";
container tunnels {
  description
  "Tunnels TE configuration data container";

  list tunnel {
    key "name";
    description "P2P TE tunnels list.";
    uses tunnel-p2p-properties;
    uses tunnel-actions;
    uses tunnel-protection-actions;
  }

  list tunnel-p2mp {
    key "name";
    unique "identifier";
    description "P2MP TE tunnels list.";
    uses tunnel-p2mp-properties;
  }
}

/* TE LSPs ephemeral state container data */
grouping lsp-properties-state {
  description
  "LSPs state operational data grouping";
  leaf source {
    type te-types:te-node-id;
    description
    "Tunnel sender address extracted from
    SESSION object";
    reference "RFC3209";
  }

  leaf destination {
    type te-types:te-node-id;
    description
    "Tunnel endpoint address extracted from
    SESSION object";
    reference "RFC3209";
  }

  leaf tunnel-id {
    type uint16;
    description
    "Tunnel identifier used in the SESSION
    that remains constant over the life
    of the tunnel.";
    reference "RFC3209";
  }
}
leaf lsp-id {
  type uint16;
  description "Identifier used in the SENDER_TEMPLATE
  and the FILTER_SPEC that can be changed
to allow a sender to share resources with
itself.";
  reference "RFC3209";
}
leaf extended-tunnel-id {
  type yang:dotted-quad;
  description "Extended Tunnel ID of the LSP.";
  reference "RFC3209";
}
leaf operational-state {
  type identityref {
    base te-types:lsp-state-type;
  }
  description "LSP operational state.";
}
leaf path-setup-protocol {
  type identityref {
    base te-types:path-signaling-type;
  }
  default te-types:path-setup-static;
  description "Signaling protocol used to set up this tunnel";
}
leaf origin-type {
  type enumeration {
    enum ingress {
      description "Origin ingress";
    }
    enum egress {
      description "Origin egress";
    }
    enum transit {
      description "transit";
    }
  }
  default 'ingress';
  description "Origin type of LSP relative to the location
  of the local switch in the path.";
}
leaf lsp-resource-status {
  type enumeration {
    enum primary {
      description
      "A primary LSP is a fully established LSP for which the resource allocation has been committed at the data plane";
    }
    enum secondary {
      description
      "A secondary LSP is an LSP that has been provisioned in the control plane only; e.g. resource allocation has not been committed at the data plane";
    }
  }
  default 'primary';
  description "LSP resource allocation type";
  reference "RFC4872, section 4.2.1";
}

uses protection-restoration-properties-state;
}

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

/**
 * TE configurations container
 */
container te {
  presence "Enable TE feature.";
  description "TE global container.";

  /* TE Global Configuration Data */
  uses globals-grouping;

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

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

/* TE Global RPCs/execution Data */
rpc globals-rpc {
  description
"Execution data for TE global."
}

/* TE interfaces RPCs/execution Data */
rpc interfaces-rpc {
  description
  "Execution data for TE interfaces.";
}

/* TE Tunnel RPCs/execution Data */
rpc tunnels-rpc {
  description "TE tunnels RPC nodes";
  input {
    container tunnel-info {
      description "Tunnel Identification";
      choice type {
        description "Tunnel information type";
        case tunnel-p2p {
          leaf p2p-id {
            type tunnel-ref;
            description "P2P TE tunnel";
          }
        }
        case tunnel-p2mp {
          leaf p2mp-id {
            type tunnel-p2mp-ref;
            description "P2MP TE tunnel";
          }
        }
      }
    }
    output {
      container result {
        description
        "The container result of the RPC operation";
        leaf result {
          type enumeration {
            enum success {
              description "Origin ingress";
            }
            enum in-progress {
              description "Origin egress";
            }
            enum fail {
              description "transit";
            }
          }
        }
      }
    }
  }
}
Internet-Draft             TE YANG Data Model                 April 2019

description "The result of the RPC operation";
}
}
}
}

/* TE Global Notification Data */
notification globals-notif {
  description
    "Notification messages for Global TE."
}

/* TE Tunnel Notification Data */
notification tunnels-notif {
  description
    "Notification messages for TE tunnels."
}
}<CODE ENDS>

Figure 7: TE generic YANG module

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

- ietf-yang-types and ietf-inet-types defined in [RFC6991]
- ietf-interfaces defined in [RFC8343]
- ietf-routing-types defined in [RFC8294]
- ietf-te-types defined in [I-D.ietf-teas-yang-te-types]
- ietf-te defined in this document

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


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

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

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

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

import ietf-routing-types {
    prefix "rt-types";
    reference "RFC8294: Common YANG Data Types for the Routing Area";
}

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

contact
    "WG Web: <http://tools.ietf.org/wg/teas/>
    WG List: <mailto:teas@ietf.org>
    WG Chair: Lou Berger
               <mailto:lberger@labn.net>
    WG Chair: Vishnu Pavan Beeram
               <mailto:vbeeram@juniper.net>
    Editor: Tarek Saad
            <mailto:tsaad@juniper.net>
    Editor: Rakesh Gandhi
            <mailto:rgandhi@cisco.com>
    Editor: Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
**TE LSP device state grouping**

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

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

"YANG data module for TE device configurations, state, RPC and notifications.";
"lsp expiration delay time";
}
}

container downstream-info {
  when ".*/te:origin-type != 'egress'" {  
    description "Applicable to ingress LSPs only";
  }
  description  
  "downstream information";

  leaf nhop {  
    type inet:ip-address;  
    description  
    "downstream nexthop.";
  }

  leaf outgoing-interface {  
    type if:interface-ref;  
    description  
    "downstream interface.";
  }

  leaf neighbor {  
    type inet:ip-address;  
    description  
    "downstream neighbor.";
  }

  leaf label {  
    type rt-types:generalized-label;  
    description  
    "downstream label.";
  }
}

container upstream-info {
  when ".*/te:origin-type != 'ingress'" {  
    description "Applicable to non-ingress LSPs only";
  }
  description  
  "upstream information";

  leaf phop {  
    type inet:ip-address;  
    description  
    "upstream nexthop or previous-hop.";
  }
}
leaf neighbor {
    type inet:ip-address;
    description "upstream neighbor."
}

leaf label {
    type rt-types:generalized-label;
    description "upstream label."
}

/**
 * Device general groupings.
 */
grouping tunnel-device-config {
    description "Device TE tunnel configs";
    leaf path-invalidation-action {
        type identityref {
            base te-types:path-invalidation-action-type;
        }
        description "Tunnel path invalidation action"
    }
}

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

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

/**
 * TE interface device generic groupings
 */
grouping te-admin-groups-config {
    description
        "TE interface affinities grouping";
    choice admin-group-type {
        description
            "TE interface administrative groups representation type";
        case value-admin-groups {
            choice value-admin-group-type {
                description "choice of admin-groups";
                case admin-groups {
                    description
                        "Administrative group/Resource class/Color.";
                    leaf admin-group {
                        type te-types:admin-group;
                        description
                            "TE interface administrative group";
                    }
                }
            }
        }
    }
}
case extended-admin-groups {
    if-feature te-types:extended-admin-groups;
    description "Extended administrative group/Resource
class/Color.";
    leaf extended-admin-group {
        type te-types:extended-admin-group;
        description "TE interface extended administrativei
group";
    }
}
}
}
}
}
}
}

/* 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";
                    }}}}
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.";
    }
  }
}
**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
"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";
}
"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."

```yamel

type rt-types:percentage;
description
"The thresholds (expressed as a percentage of the maximum reservable bandwidth of the interface) at which bandwidth updates are flooded - used both when the bandwidth is increasing and decreasing";

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

/* TE interface switching capabilities */
grouping te-switching-cap-config {
    description "TE interface switching capabilities";
    list switching-capabilities {
        key "switching-capability";
        description "List of interface capabilities for this interface";
        leaf switching-capability {
            type identityref {
                base te-types:switching-capabilities;
            }
            description "Switching Capability for this interface";
        }
        leaf encoding {
            type identityref {
                base te-types:lsp-encoding-types;
            }
            description "Encoding supported by this interface";
        }
    }
}
```
grouping te-advertisements-state {
  description
  "TE interface advertisements state grouping";
  container te-advertisements-state {
    description
    "TE interface advertisements state container";
    leaf flood-interval {
      type uint32;
      description
      "The periodic flooding interval";
    }
    leaf last-flooded-time {
      type uint32;
      units seconds;
      description
      "Time elapsed since last flooding in seconds";
    }
    leaf next-flooded-time {
      type uint32;
      units seconds;
      description
      "Time remained for next flooding in seconds";
    }
    leaf last-flooded-trigger {
      type enumeration {
        enum link-up {
          description "Link-up flooding trigger";
        }
        enum link-down {
          description "Link-up flooding trigger";
        }
        enum threshold-up {
          description
          "Bandwidth reservation up threshold";
        }
        enum threshold-down {
          description
          "Bandwidth reservation down threshold";
        }
        enum bandwidth-change {
          description "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 advertized-level-areas {
    key level-area;
    description "List of areas the TE interface is advertised in";
    leaf level-area {
        type uint32;
        description "The IGP area or level where the TE interface state is advertised in";
    }
}

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

grouping te-all-attributes {
    description "TE attributes configuration grouping for all interfaces";
    uses te-igp-flooding-bandwidth-config;
} /*** End of TE interfaces device groupings ***/

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

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

/* TE tunnels device configuration augmentation */
augment "/te:te/te:tunnels/te:tunnel" {
    description
        "Tunnel device dependent augmentation";
    uses lsp-device-timers-config;
}

/* TE LSPs device state augmentation */
augment "/te:te/te:lsps-state/te:lsp" {
    description
        "LSP device dependent augmentation";
    uses lsps-device-state;
}
augment "/te:te/te:tunnels/te:tunnel/te:p2p-secondary-paths" + 
    "/te:p2p-secondary-path/te:lsps/te:lsp" {
    description
        "LSP device dependent augmentation";
    uses lsps-device-state;
}
augment "/te:te/te:tunnels/te:tunnel/te:p2p-primary-paths" + 
    "/te:p2p-primary-path/te:lsps/te:lsp" {
    description
        "LSP device dependent augmentation";
    uses lsps-device-state;
}

/* TE interfaces RPCs/execution Data */
rpc interfaces-rpc {
  description
    "Execution data for TE interfaces.";
}

/* TE Interfaces Notification Data */
notification interfaces-notif {
  description
    "Notification messages for TE interfaces.";
}

<CODE ENDS>

Figure 8: TE device specific YANG module

5. IANA Considerations

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

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

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
prefix:     ietf-te
reference:  RFCXXXX

name:       ietf-te-device
prefix:     ietf-te-device
reference:  RFCXXXX

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC8341] provides means to restrict access for particular NETCONF
users to a pre-configured subset of all available NETCONF protocol operations and content.

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

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

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

"/te/lsps-state": This list specifies the state derived LSPs. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

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

7. Acknowledgement

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

The authors would also like to thank Loa Andersson, Lou Berger, Sergio Belotti, Italo Busi, Carlo Perocchio, Francesco Lazzeri, Aihua Guo, Dhruv Dhody, Anurag Sharma, and Xian Zhang for their comments and providing valuable feedback on this document.

8. Contributors
9. References

9.1. Normative References

[I-D.ietf-teas-yang-path-computation]

[I-D.ietf-teas-yang-rsvp]

[I-D.ietf-teas-yang-te-types]


9.2. Informative References


Authors’ Addresses

Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

Rakesh Gandhi
Cisco Systems Inc
Email: rgandhi@cisco.com

Xufeng Liu
Volta Networks
Email: xufeng.liu.ietf@gmail.com
Vishnu Pavan Beeram  
Juniper Networks  
Email: vbeeram@juniper.net

Igor Bryskin  
Huawei Technologies  
Email: Igor.Bryskin@huawei.com
YANG Data Model for Traffic Engineering (TE) Topologies
draft-ietf-teas-yang-te-topo-22

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

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

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

Copyright Notice

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

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

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

Conventions used in this document

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

Table of Contents

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

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

This document defines a YANG [RFC7950] data model for representing and manipulating TE Topologies. This model contains technology
agnostic TE Topology building blocks that can be augmented and used by other technology-specific TE Topology models.

1.1. Terminology

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

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

Some of the key terms used in this document are:

- TED: The Traffic Engineering Database is a collection of all TE information about all TE nodes and TE links in a given network.
- TE-Topology: The TE Topology is a schematic arrangement of TE nodes and TE links in a given TED. It forms the basis for a graph suitable for TE path computations.
- Native TE Topology: Native TE Topology is a topology that is native to a given provider network. Native TE topology could be discovered via various routing protocols and/or subscribe/publish techniques. This is the topology on which path computational algorithms are run to compute TE Paths.
- Customized TE Topology: Customized TE Topology is a custom topology that is produced by a provider for a given client. This topology typically makes abstractions on the provider’s Native TE Topology, and is provided to the client. The client receives the Customized TE Topology, and merges it into the client’s Native TE Topology. The client’s path computational algorithms aren’t typically run on the Customized TE Topology; they are run on the client’s Native TE Topology after the merge.

1.2. Tree Structure

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

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>YANG module</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang</td>
<td>ietf-yang-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>inet</td>
<td>ietf-inet-types</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>nw</td>
<td>ietf-network</td>
<td>[RFC6991]</td>
</tr>
<tr>
<td>nt</td>
<td>ietf-network-topology</td>
<td>[RFC8345]</td>
</tr>
<tr>
<td>te-types</td>
<td>ietf-te-types</td>
<td>[I-D.ietf-teas-yang-te-types]</td>
</tr>
</tbody>
</table>

Table 1: Prefixes and corresponding YANG modules

2. Characterizing TE Topologies

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

- TE Topology is an abstract control-plane representation of the data-plane topology. Hence attributes specific to the data-plane must make their way into the corresponding TE Topology modeling. The TE Topology comprises of dynamic auto-discovered data as well as fairly static data associated with data-plane nodes and links. The dynamic data may change frequently, such as unreserved bandwidth available on data-plane links. The static data rarely changes, such as layer network identification, switching and adaptation capabilities and limitations, fate sharing, and administrative colors. It is possible for a single TE Topology to encompass TE information at multiple switching layers.

- TE Topologies are protocol independent. Information about topological elements may be learnt via link-state protocols, but the topology can exist without being dependent on any particular protocol.

- TE Topology may not be congruent to the routing topology in a given TE System. The routing topology is constructed based on routing adjacencies. There isn’t always a one-to-one association between a TE-link and a routing adjacency. For example, the presence of a TE link between a pair of nodes doesn’t necessarily imply the existence of a routing-adjacency between these nodes. To
learn more, see [I-D.ietf-teas-te-topo-and-tunnel-modeling] and [I-D.ietf-teas-yang-l3-te-topo].

- Each TE Topological element has at least one information source associated with it. In some scenarios, there could be more than one information source associated with any given topological element.

- TE Topologies can be hierarchical. Each node and link of a given TE Topology can be associated with respective underlay topology. This means that each node and link of a given TE Topology can be associated with an independent stack of supporting TE Topologies.

- TE Topologies can be customized. TE topologies of a given network presented by the network provider to its client could be customized on per-client request basis. This customization could be performed by provider, by client or by provider/client negotiation. The relationship between a customized topology and provider’s native topology could be captured as hierarchical (overlay-underlay), but otherwise the two topologies are decoupled from each other. A customized topology is presented to the client, while provider’s native topology is known in its entirety to the provider itself.
3. Modeling Abstractions and Transformations

![Diagram of TE Topology Modeling Abstractions](image)

Figure 1: TE Topology Modeling Abstractions

3.1. TE Topology

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

3.2. TE Node

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

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

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

3.3. TE Link

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

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

3.4. Transitional TE Link for Multi-Layer Topologies

Networks are typically composed of multiple network layers where one or multiple signals in the client layer network can be multiplexed and encapsulated into a server layer signal [RFC5212] [G.805]. The server layer signal can be carried in the server layer network across multiple nodes until the server layer signal is terminated and the client layer signals reappear in the node that terminates the server layer signal. Examples of multi-layer networks are: IP over MPLS over Ethernet, low order Optical Data Unit-k (ODUk) signals multiplexed into a high order ODU1 (l>k) carried over an Optical Channel (OCh) signal in an optical transport network as defined in [G.872] and [G.709].
TE links as defined in Section 3.3. can be used to represent links within a network layer. In case of a multi-layer network, TE nodes and TE links only allow representation of each network layer as a separate TE topology. Each of these single layer TE topologies would be isolated from their client and their server layer TE topology, if present. The highest and the lowest network layer in the hierarchy only have a single adjacent layer below or above, respectively. Multiplexing of client layer signals and encapsulating them into a server layer signal requires a function that is provided inside a node (typically realized in hardware). This function is also called layer transition.

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

When the server layer trail has been established between the server layer LTP of two transitional links in the server layer network, the resulting client layer link in the data plane will be represented as a normal TE link in the client layer topology. The transitional TE links will re-appear when the server layer trail has been torn down.
3.5. TE Link Termination Point (LTP)

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

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

3.6. TE Tunnel Termination Point (TTP)

TE tunnel termination point (TTP) is an element of TE topology representing one or several of potential transport service termination points (i.e. service client adaptation points such as
WDM/OCh transponder). TTP is associated with (hosted by) exactly one TE node. TTP is assigned a unique ID within the TE node scope. Depending on the TE node’s internal constraints, a given TTP hosted by the TE node could be accessed via one, several or all TE links terminated by the TE node.

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

3.7. TE Node Connectivity Matrix

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

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

3.8. TTP Local Link Connectivity List (LLCL)

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

In Figure 1, the LLCL on Node-1 is:
\{<TTP-1, LTP-5>, <TTP-1, LTP-2>, <TTP-2, LTP-3>, <TTP-2, LTP-4>\}

3.9. TE Path

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

In Figure 1, the TE Path for TE-Tunnel-1 is:
(Node-1:TTP-1, Link-12, Node-2, Link-23, Node-3:TTP1)
3.10. TE Inter-Layer Lock

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

```
+---+          __
|   | TE Node  / TTP  o LTP
+---+

----- TE Link
***** TTP Local Link Connectivity
```

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

Figure 3: TE Inter-Layer Lock ID Associations

On the picture above a TE inter-layer lock with IL_1 ID associates 6 client layer LTPs (C-LTP-1 - C-LTP-6) with two server layer TTPs (S-TTP-1 and S-TTP-2). They all have the same attribute - TE inter-layer lock ID: IL-1, which is the only thing that indicates the association. A given LTP may have 0, 1 or more inter-layer lock IDs. In the latter case this means that the data arriving at the LTP may be adopted onto any of TTPs associated with all specified inter-layer locks. For example, C-LTP-1 could have two inter-layer lock IDs - IL-1 and IL-2. This would mean that C-LTP-1 for adaptation purposes could use not just TTPs associated with inter-layer lock IL-1 (i.e.
S-TTP-1 and S-TTP-2 on the picture), but any of TTPs associated with inter-layer lock IL-2 as well. Likewise, a given TTP may have one or more inter-layer lock IDs, meaning that it can offer the adaptation service to any of client layer LTPs with inter-layer lock ID matching one of its own. Additionally, each TTP has an attribute - Unreserved Adaptation Bandwidth, which announces its remaining adaptation resources sharable between all potential client LTPs.

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

3.11. Underlay TE topology

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

3.12. Overlay TE topology

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

3.13. Abstract TE topology

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

Figure 4 shows an example of abstract TE topology.

4. Model Applicability

4.1. Native TE Topologies

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

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

![Figure 6a: Example Network Topology](image)

![Figure 6b: Native TE Topologies as seen on Node R3](image)

4.2. Customized TE Topologies

Customized TE topology is a topology that was modified by the provider to honor a particular client’s requirements or preferences. The model discussed in this draft can be used to represent, retrieve and manipulate customized TE Topologies. The model allows the provider to present the network in abstract TE Terms on a per client...
These customized topologies contain sufficient information for the path computing client to select paths according to its policies.

Consider the network topology depicted in Figure 7. This is a typical packet optical transport deployment scenario where the WDM layer network domain serves as a Server Network Domain providing transport connectivity to the packet layer network Domain (Client Network Domain). Nodes R1, R2, R3 and R4 are IP routers that are connected to an Optical WDM transport network. A, B, C, D, E and F are WDM nodes that constitute the Server Network Domain.
Figure 8a: Paths within the provider domain

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

Figure 8b: Customized TE Topology provided to the Client

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

[R1] +++++++ [A] ++++++++++++++++++++ [E] +++++++ [R3]
++++
++++
++++

[R2] +++++++ [B] ++++++++++++++++++++ [F] +++++++ [R4]

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

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

A customized TE topology is not necessarily an abstract TE topology. The provider may produce, for example, an abstract TE topology of certain type (e.g. single-abstract-node-with-connectivity-matrix topology, a border-nodes-connected-via-mesh-of-abstract-links topology, etc.) and expose it to all/some clients in expectation that the clients will use it without customization. On the other hand, a client may request a customized version of the provider’s native TE topology (e.g. by requesting removal of TE links...
which belong to certain layers, are too slow, not protected and/or have a certain affinity). Note that the resulting TE topology will not be abstract (because it will not contain abstract elements), but customized (modified upon client’s instructions).

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

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

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

4.3. Merging TE Topologies Provided by Multiple Providers

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

In particular, the process of merging TE topologies includes:

- Identifying neighboring domains and locking their topologies horizontally by connecting their inter-domain open-ended TE links;
- Renaming TE node, link, and SRLG IDs to ones allocated from a separate name space; this is necessary because all TE topologies are considered to be, generally speaking, independent with a possibility of clashes among TE node, link or SRLG IDs;
- Locking, vertically, TE topologies associated with different layer networks, according to provided topology inter-layer locks; this is to facilitate inter-layer path computations across multiple TE topologies provided by the same topology provider.
Figure 9 illustrates the process of merging, by the client, of TE topologies provided by the client’s providers. In the Figure, each of the two providers caters to the client (abstract or native) TE topology, describing the network domain under the respective provider’s control. The client, by consulting the attributes of the inter-domain TE links - such as inter-domain plug IDs or remote TE node/link IDs (as defined by the TE Topology model) - is able to determine that:

a) the two domains are adjacent and are inter-connected via three inter-domain TE links, and;
b) each domain is connected to a separate customer site, connecting the left domain in the Figure to customer devices C-11 and C-12, and the right domain to customer devices C-21, C-22 and C-23.

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

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

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

Once the merging process is complete, the client can use the merged TE topology for path computations across both domains, for example, to compute a TE path connecting C-11 to C-23.
4.4. Dealing with Multiple Abstract TE Topologies Provided by the Same Provider

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

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

Note that allowing for the client mix-n-matching of multiple TE topologies assumes that inter-domain plug IDs (rather than remote nodeID/linkID) option is used for identifying neighboring domains and inter-domain TE link resolution.
Client’s Merged TE Topology 2

Client’s Merged TE Topology 3

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

It is important to note that each of the three native (merged) TE topologies could be used by the client for computing TE paths for any of the multi-domain services. The choice as to which topology to use for a given service depends on the service parameters/requirements and the topology’s style, optimization criteria and the level of details.
5. Modeling Considerations

5.1. Network topology building blocks

The network topology building blocks are discussed in [RFC8345]. The TE Topology model proposed in this document augments and uses the ietf-network-topology module defined in [RFC8345].

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

V

+------------------------+
| TE Topology Model      |
+------------------------+
```

Figure 12: Augmenting the Network Topology Model

5.2. Technology agnostic TE Topology model

The TE Topology model proposed in this document is meant to be network technology agnostic. Other technology specific TE Topology models can augment and use the building blocks provided by the proposed model.
Figure 13: Augmenting the Technology agnostic TE Topology model

5.3. Model Structure

The high-level model structure proposed by this document is as shown below:

```
module: ietf-te-topology
augment /nw:networks/nw:network/nw:network-types:
  +--rw te-topology!

augment /nw:networks:
  +--rw te!
    +--rw templates
      +--rw node-template* [name] {template}?
        | ................
      +--rw link-template* [name] {template}?
        ................

augment /nw:networks/nw:network:
  +--rw te-topology-identifier
    | +--rw provider-id? te-global-id
    | +--rw client-id? te-global-id
    | +--rw topology-id? te-topology-id
    +--rw te!
      | ................

augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id? te-types:te-node-id
  +--rw te!
    | ................
    +--rw tunnel-termination-point* [tunnel-tp-id]
```
5.4. Topology Identifiers

The TE-Topology is uniquely identified by a key that has 3 constituents - topology-id, provider-id and client-id. The combination of provider-id and topology-id uniquely identifies a native TE Topology on a given provider. The client-id is used only when Customized TE Topologies come into play; a value of "0" is used as the client-id for native TE Topologies.

5.5. Generic TE Link Attributes

The model covers the definitions for generic TE Link attributes - bandwidth, admin groups, SRLGs, switching capabilities, TE metric extensions etc.
5.6. Generic TE Node Attributes

The model covers the definitions for generic TE Node attributes.

The definition of a generic connectivity matrix is shown below:

```
++--rw te-node-attributes
    ...........
    ++--rw connectivity-matrices
        ...........
        | +--rw connectivity-matrix* [id]
        |    ++--rw id          uint32
        |    ++--rw from
        |        | +--rw tp-ref?    leafref
        |        | +--rw label-restrictions
        |    ++--rw to
        |        | +--rw tp-ref?    leafref
        |        | +--rw label-restrictions
        |    ++--rw is-allowed? boolean
        |    | +--rw underlay! {te-topology-hierarchy}? 
        |    | .........
        |    | +--rw path-constraints
        |    | .........
        |    | +--rw optimizations
        |    | .........
        |    | +--ro path-properties
        |    | .........
```

The definition of a TTP Local Link Connectivity List is shown below:

```
++--rw tunnel-termination-point* [tunnel-tp-id]
    ++--rw tunnel-tp-id          binary
    ++--rw admin-status?        te-types:te-admin-status
    ++--rw name?                string
    ++--rw switching-capability? identityref
    ++--rw encoding?            identityref
    ++--rw inter-layer-lock-id* uint32
```
The attributes directly under container connectivity-matrices are the default attributes for all connectivity-matrix entries when the per entry corresponding attribute is not specified. When a per entry attribute is specified, it overrides the corresponding attribute directly under the container connectivity-matrices. The same rule applies to the attributes directly under container local-link-connectivities.

Each TTP (Tunnel Termination Point) MAY be supported by one or more supporting TTPs. If the TE node hosting the TTP in question refers to a supporting TE node, then the supporting TTPs are hosted by the supporting TE node. If the TE node refers to an underlay TE topology, the supporting TTPs are hosted by one or more specified TE nodes of the underlay TE topology.

5.7. TED Information Sources

The model allows each TE topological element to have multiple TE information sources (OSPF-TE, ISIS-TE, BGP-LS, User-Configured, System-Processed, Other). Each information source is associated with a credibility preference to indicate precedence. In scenarios where a customized TE Topology is merged into a Client’s native TE Topology, the merged topological elements would point to the corresponding customized TE Topology as its information source.
augment /nw:networks/nw:network/nw:node:
  +--rw te!
    ...........
    +--ro information-source?  te-info-source
    +--ro information-source-instance?  string
    +--ro information-source-state
      +--ro credibility-preference?  uint16
      +--ro logical-network-element?  string
      +--ro network-instance?  string
      +--ro topology
        +--ro node-ref?  leafref
        +--ro network-ref?  leafref
      +--ro information-source-entry*
        [information-source information-source-instance]
        +--ro information-source  te-info-source
        +--ro information-source-instance  string
        ...........

augment /nw:networks/nw:network/nt:link:
  +--rw te!
    ...........
    +--ro information-source?  te-info-source
    +--ro information-source-instance?  string
    +--ro information-source-state
      +--ro credibility-preference?  uint16
      +--ro logical-network-element?  string
      +--ro network-instance?  string
      +--ro topology
        +--ro link-ref?  leafref
        +--ro network-ref?  leafref
      +--ro information-source-entry*
        [information-source information-source-instance]
        +--ro information-source  te-info-source
        +--ro information-source-instance  string
        ...........

5.8. Overlay/Underlay Relationship

The model captures overlay and underlay relationship for TE nodes/links. For example - in networks where multiple TE Topologies are built hierarchically, this model allows the user to start from a specific topological element in the top most topology and traverse all the way down to the supporting topological elements in the bottom most topology.

This relationship is captured via the "underlay-topology" field for the node and via the "underlay" field for the link. The use of these
fields is optional and this functionality is tagged as a "feature" ("te-topology-hierarchy").

augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?   te-types:te-node-id
  +--rw te!
     +--rw te-node-template*   leafref {template}?
     +--rw te-node-attributes
        |  +--rw admin-status?    te-types:te-admin-status
        |     ........................
     +--rw underlay-topology {te-topology-hierarchy}?
        +--rw network-ref?   leafref

augment /nw:networks/nw:network/nt:link:
  +--rw te!
     +--rw te-link-attributes
        ........................
     +--rw underlay {te-topology-hierarchy}?
        |  +--rw enabled?    boolean
        |  +--rw primary-path
        |     +--rw network-ref?   leafref
        |     ........................
        |  +--rw backup-path* [index]
        |     |  +--rw index     uint32
        |     |  +--rw network-ref?   leafref
        |     ........................
        |  +--rw protection-type?   identityref
        +--rw tunnel-termination-points
           |  +--rw source?    binary
           |  +--rw destination?  binary
           +--rw tunnels
              ........................

5.9. Templates

The data model provides the users with the ability to define templates and apply them to link and node configurations. The use of "template" configuration is optional and this functionality is tagged as a "feature" ("template").

augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?   te-types:te-node-id
  +--rw te!
     +--rw te-node-template*   leafref {template}?
     -> ../../te/templates/node-template/name
Multiple templates can be specified to a configuration element. When two or more templates specify values for the same configuration field, the value from the template with the highest priority is used. The range of the priority is from 0 to 65535, with a lower number indicating a higher priority. The reference-change-policy specifies the action that needs to be taken when the template changes on a configuration element that has a reference to this template. The choices of action include taking no action, rejecting the change to the template and applying the change to the corresponding configuration.

5.10. Scheduling Parameters

The model allows time scheduling parameters to be specified for each topological element or for the topology as a whole. These parameters allow the provider to present different topological views to the client at different time slots. The use of "scheduling parameters" is optional.

The YANG data model for configuration scheduling is defined in [I-D.liu-netmod-yang-schedule], which allows specifying configuration schedules without altering this data model.
5.11. Notifications

Notifications are a key component of any topology data model.

[I-D.ietf-netconf-subscribed-notifications] and
[I-D.ietf-netconf-yang-push] define a subscription and push mechanism
for YANG datastores. This mechanism currently allows the user to:

- Subscribe notifications on a per client basis
- Specify subtree filters or xpath filters so that only interested
  contents will be sent.
- Specify either periodic or on-demand notifications.

6. Guidance for Writing Technology Specific TE Topology Augmentations

The TE topology model defined in this document is technology agnostic
as it defines concepts, abstractions and attributes that are common
across multiple network technologies. It is envisioned that this base
model will be widely used when defining technology specific TE
topology models for various layer networks.
[I-D.ietf-ccamp-wson-yang], [I-D.ietf-ccamp-otn-topo-yang], and
[I-D.ietf-teas-yang-l3-te-topo] are some examples of technology
specific TE Topology models. Writers of such models are encouraged to
augment the basic TE topology model’s containers, such as TE
Topology, TE Node, TE Link, Link Termination Point (LTP), Tunnel
Termination Point (TTP), Bandwidth and Label with the layer specific
attributes instead of defining new containers.

Consider the following technology specific example-topology model:

module: example-topology
    augment /nw:networks/nw:network/nw:network-types/tet:te-topology:
        ---rw example-topology!
    augment /nw:networks/nw:network/tet:te:
        ---rw attributes
            ---rw attribute-1?  uint8
    augment /nw:networks/nw:network/nw:node/tet:te
        /tet:te-node-attributes:
            ---rw attributes
                ---rw attribute-2?  uint8
    augment /nw:networks/nw:network/nw:node/tet:te
        /tet:te-node-attributes/tet:connectivity-matrices:
            ---rw attributes
                ---rw attribute-3?  uint8
    augment /nw:networks/nw:network/nw:node/tet:te
The technology specific TE bandwidth for this example topology can be specified using the following augment statements:

```
  /tet:te-link-attributes
  /tet:interface-switching-capability/tet:max-lsp-bandwidth
  /tet:te-bandwidth/tet:technology:
  +--:(example)
    +--rw example
    +--rw bandwidth-1? uint32

  /tet:te-link-attributes/tet:max-link-bandwidth
  /tet:te-bandwidth/tet:technology:
  +--:(example)
    +--rw example
    +--rw bandwidth-1? uint32

  /tet:te-link-attributes/tet:max-resv-link-bandwidth
  /tet:te-bandwidth/tet:technology:
  +--:(example)
    +--rw example
    +--rw bandwidth-1? uint32
```

Liu, et al            Expires December 19, 2019   [Page 34]
/tet:te-link-attributes/tet:unreserved-bandwidth
/tet:te-bandwidth/tet:technology:
  +-:(example)
  |  +-rw example
  |     +-rw bandwidth-1?  uint32
  |     augment /nw:networks/nw:network/nw:node/tet:te
  |           /tet:te-node-attributes/tet:connectivity-matrices
  |           /tet:path-constraints/tet:te-bandwidth/tet:technology:
  |     +-:(example)
  |     |  +-rw example
  |     |     +-rw bandwidth-1?  uint32
  |     |     augment /nw:networks/nw:network/nw:node/tet:te
  |     |           /tet:te-node-attributes/tet:connectivity-matrices
  |     |           /tet:connectivity-matrix/tet:path-constraints
  |     |           /tet:te-bandwidth/tet:technology:
  |     +-:(example)
  |     |  +-rw example
  |     |     +-rw bandwidth-1?  uint32
  |     |     augment /nw:networks/nw:network/nw:node/tet:te
  |     |           /tet:information-source-entry/tet:connectivity-matrices
  |     |           /tet:path-constraints/tet:te-bandwidth/tet:technology:
  |     +-:(example)
  |     |  +-ro example
  |     |     +-ro bandwidth-1?  uint32
  |     |     augment /nw:networks/nw:network/nw:node/tet:te
  |     |           /tet:information-source-entry/tet:connectivity-matrices
  |     |           /tet:connectivity-matrix/tet:path-constraints
  |     |           /tet:te-bandwidth/tet:technology:
  +-:(example)
     +-ro example
     +-ro bandwidth-1?  uint32
     augment /nw:networks/nw:network/nw:node/tet:te
     /tet:tunnel-termination-point/tet:client-layer-adaptation
     /tet:switching-capability/tet:te-bandwidth
     /tet:technology:
  +-:(example)
     +-rw example
     +-rw bandwidth-1?  uint32
     augment /nw:networks/nw:network/nw:node/tet:te
     /tet:tunnel-termination-point
     /tet:local-link-connectivities/tet:path-constraints
augment /nw:networks/nw:network/nw:node/tet:te
/tet:tunnel-termination-point
/tet:local-link-connectivities
/tet:local-link-connectivity/tet:path-constraints
/tet:te-bandwidth/tet:technology:

---:(example)
---rw example
  ---rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:te-link-attributes
/tet:interface-switching-capability/tet:max-lsp-bandwidth
/tet:te-bandwidth/tet:technology:

---:(example)
---rw example
  ---rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:te-link-attributes/tet:max-link-bandwidth
/tet:te-bandwidth/tet:technology:

---:(example)
---rw example
  ---rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:te-link-attributes/tet:max-resv-link-bandwidth
/tet:te-bandwidth/tet:technology:

---:(example)
---rw example
  ---rw bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:information-source-entry
/tet:interface-switching-capability/tet:max-lsp-bandwidth
/tet:te-bandwidth/tet:technology:

---:(example)
---ro example
  ---ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:information-source-entry/tet:max-link-bandwidth
/tet:te-bandwidth/tet:technology:
++--: (example)
++--ro example
++--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:information-source-entry/tet:max-resv-link-bandwidth
/tet:te-bandwidth/tet:technology:
++--: (example)
++--ro example
++--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
/tet:information-source-entry/tet:unreserved-bandwidth
/tet:te-bandwidth/tet:technology:
++--: (example)
++--ro example
++--ro bandwidth-1?  uint32
augment /nw:networks/nw:network/nt:termination-point/tet:te
/tet:interface-switching-capability/tet:max-lsp-bandwidth
/tet:te-bandwidth/tet:technology:
++--: (example)
++--rw example
++--rw bandwidth-1?  uint32

The technology specific TE label for this example topology can be specified using the following augment statements:

/tet:te-link-attributes/tet:underlay/tet:primary-path
/tet:path-element/tet:type/tet:label/tet:label-hop
/tet:te-label/tet:technology:
++--: (example)
++--rw example
++--rw label-1?  uint32
/tet:te-link-attributes/tet:underlay/tet:backup-path
/tet:path-element/tet:type/tet:label/tet:label-hop
/tet:te-label/tet:technology:
++--: (example)
++--rw example
++--rw label-1?  uint32

Liu, et al Expires December 19, 2019 [Page 37]
/tet:te-link-attributes/tet:label-restrictions
/tet:label-restriction/tet:label-start/tet:te-label/tet:technology:
  +--:(example)
  +--rw example
  +--rw label-1?  uint32
  /tet:te-link-attributes/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label/tet:technology:
  +--:(example)
  +--rw example
  +--rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-end/tet:te-label/tet:technology:
  +--:(example)
  +--rw example
  +--rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label-restrictions/tet:label-restriction
  /tet:label-end/tet:te-label/tet:technology:
  +--:(example)
  +--rw example
  +--rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
  +--:(example)
  +--rw example
  +--rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:label/tet:label-hop/tet:te-label/tet:technology:
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:path-properties/tet:path-route-objects
  /tet:te-label/tet:technology:
  +-:(example)
    +-ro example
    +-ro label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:from/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
  +-:(example)
    +-rw example
    +-rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:from/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
  +-:(example)
    +-rw example
    +-rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:to/tet:label-restrictions
  /tet:label-restriction/tet:label-start/tet:te-label
  /tet:technology:
  +-:(example)
    +-rw example
    +-rw label-1?  uint32
  augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
  /tet:connectivity-matrix/tet:to/tet:label-restrictions
  /tet:label-restriction/tet:label-end/tet:te-label
  /tet:technology:
  +-:(example)
    +-rw example
    +-rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
  /tet:te-node-attributes/tet:connectivity-matrices
augment /nw:networks/nw:network/nw:node/tet:te
/tet:te-node-attributes/tet:connectivity-matrices
/tet:path-element/tet:type/tet:label/tet:label-hop
/tet:te-label/tet:technology:

++--:(example)
  +++rw example
    +++rw label-1?  uint32

augment /nw:networks/nw:network/nw:node/tet:te
/tet:te-node-attributes/tet:connectivity-matrices
/tet:path-element/tet:type/tet:label/tet:label-hop
/tet:te-label/tet:technology:

++--:(example)
  +++rw example
    +++rw label-1?  uint32

augment /nw:networks/nw:network/nw:node/tet:te
/tet:te-node-attributes/tet:connectivity-matrices
/tet:connectivity-matrix/tet:path-properties
/tet:path-route-objects/tet:path-route-object/tet:type
/tet:label/tet:label-hop/tet:te-label/tet:technology:

++--:(example)
  +++ro example
    +++ro label-1?  uint32

augment /nw:networks/nw:network/nw:node/tet:te
/tet:information-source-entry/tet:connectivity-matrices
/tet:label-restrictions/tet:label-restriction
/tet:label-start/tet:te-label/tet:technology:

++--:(example)
  +++ro example
    +++ro label-1?  uint32

augment /nw:networks/nw:network/nw:node/tet:te
/tet:information-source-entry/tet:connectivity-matrices
/tet:label-restrictions/tet:label-restriction
/tet:label-end/tet:te-label/tet:technology:

++--:(example)
  +++ro example
    +++ro label-1?  uint32

augment /nw:networks/nw:network/nw:node/tet:te
/tet:information-source-entry/tet:connectivity-matrices
/tet:label/tet:label-hop/tet:te-label/tet:technology:

++--:(example)
++--ro example
+-ro label-1?  uint32

++--:(example)
++--ro example
+-ro label-1?  uint32

++--:(example)
++--ro example
+-ro label-1?  uint32

++--:(example)
++--ro example
+-ro label-1?  uint32

++--:(example)
++--ro example
++--ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:information-source-entry/tet:connectivity-matrices
   /tet:connectivity-matrix/tet:to/tet:label-restrictions
   /tet:label-restriction/tet:label-end/tet:te-label
   /tet:technology:
++--:(example)
++--ro example
++--ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:information-source-entry/tet:connectivity-matrices
   /tet:te-label/tet:technology:
++--:(example)
++--ro example
++--ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:information-source-entry/tet:connectivity-matrices
   /tet:te-label/tet:technology:
++--:(example)
++--ro example
++--ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:information-source-entry/tet:connectivity-matrices
   /tet:label/tet:label-hop/tet:te-label/tet:technology:
++--:(example)
++--ro example
++--ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:technology:
++--:(example)
++--rw example
++--rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities/tet:label-restrictions
   /tet:label-restriction/tet:label-end/tet:te-label
   /tet:technology:
     +++:(example)
     +---rw example
        +---rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities/tet:underlay
   /tet:primary-path/tet:path-element/tet:type/tet:label
   /tet:label-hop/tet:te-label/tet:technology:
     +++:(example)
     +---rw example
        +---rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities/tet:underlay
   /tet:backup-path/tet:path-element/tet:type/tet:label
   /tet:label-hop/tet:te-label/tet:technology:
     +++:(example)
     +---rw example
        +---rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities/tet:path-properties
   /tet:path-route-objects/tet:path-route-object/tet:type
   /tet:label/tet:label-hop/tet:te-label/tet:technology:
     +++:(example)
     +---ro example
        +---ro label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities
   /tet:local-link-connectivity/tet:label-restrictions
   /tet:label-restriction/tet:label-start/tet:te-label
   /tet:technology:
     +++:(example)
     +---rw example
        +---rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities
   /tet:local-link-connection/tet:label-restrictions
   /tet:label-restriction/tet:label-end/tet:te-label
   /tet:technology:
   +--:(example)
      +--rw example
      +--rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities
   /tet:local-link-connection/tet:underlay
   /tet:primary-path/tet:path-element/tet:type/tet:label
   /tet:label-hop/tet:te-label/tet:technology:
   +--:(example)
      +--rw example
      +--rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities
   /tet:local-link-connection/tet:underlay/tet:backup-path
   /tet:path-element/tet:type/tet:label/tet:label-hop
   /tet:te-label/tet:technology:
   +--:(example)
      +--rw example
      +--rw label-1?  uint32
augment /nw:networks/nw:network/nw:node/tet:te
   /tet:tunnel-termination-point
   /tet:local-link-connectivities
   /tet:local-link-connection/tet:path-properties
   /tet:path-route-objects/tet:path-route-object/tet:type
   /tet:label/tet:label-hop/tet:te-label/tet:technology:
   +--:(example)
      +--ro example
      +--ro label-1?  uint32
augment /nw:networks/nw:network/nt:link/tet:te
   /tet:te-link-attributes/tet:label-restrictions
   /tet:label-restriction/tet:label-start/tet:te-label
   /tet:technology:
   +--:(example)
The YANG module to implement the above example topology can be seen in Appendix C.
This module references [RFC1195], [RFC3209], [RFC3272], [RFC3471], [RFC3630], [RFC3785], [RFC4201], [RFC4202], [RFC4203], [RFC4206], [RFC4872], [RFC5152], [RFC5212], [RFC5305], [RFC5316], [RFC5329], [RFC5392], [RFC6001], [RFC6241], [RFC6991], [RFC7308], [RFC7471], [RFC7579], [RFC7752], [RFC8345], and [I-D.ietf-teas-yang-te-types].

<CODE BEGINS> file "ietf-te-topology@2019-02-07.yang"
module ietf-te-topology {
  yang-version 1.1;

  prefix "tet";

  import ietf-yang-types {
    prefix "yang";
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-inet-types {
    prefix "inet";
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-te-types {
    prefix "te-types";
    reference "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
  }

  import ietf-network {
    prefix "nw";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-network-topology {
    prefix "nt";
    reference "RFC 8345: A YANG Data Model for Network Topologies";
  }

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

contact
"WG Web: <http://tools.ietf.org/wg/teas/>
WG List: <mailto:teas@ietf.org>
Editor: Xufeng Liu <mailto:xufeng.liu.ietf@gmail.com>
Editor: Igor Bryskin <mailto:Igor.Bryskin@huawei.com>
Editor: Vishnu Pavan Beeram <mailto:vbeeram@juniper.net>
Editor: Tarek Saad <mailto:tsaad@juniper.net>
Editor: Himanshu Shah <mailto:hshah@ciena.com>
Editor: Oscar Gonzalez De Dios <mailto:oscar.gonzalezdedios@telefonica.com>"

description
"TE topology model for representing and manipulating technology agnostic TE Topologies.

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

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

This version of this YANG module is part of RFC XXXX; see the

Liu, et al Expires December 19, 2019 [Page 47]
RFC itself for full legal notices.

revision "2019-02-07" {
  description "Initial revision"
  reference "RFC XXXX: YANG Data Model for TE Topologies"
  // RFC Ed.: replace XXXX with actual RFC number and remove
  // this note
}

/*
 * Features
 */

feature nsrlg {
  description
    "This feature indicates that the system supports NSRLG
     (Not Sharing Risk Link Group)."
}

feature te-topology-hierarchy {
  description
    "This feature indicates that the system allows underlay
     and/or overlay TE topology hierarchy."
}

feature template {
  description
    "This feature indicates that the system supports
     template configuration."
}

/*
 * Typedefs
 */
typedef geographic-coordinate-degree {
  type decimal64 {
    fraction-digits 8;
  }
  description
    "Decimal degree (DD) used to express latitude and longitude
     geographic coordinates.";
} // geographic-coordinate-degree
typedef te-info-source {
  type enumeration {
    enum "unknown" {
      description "The source is unknown.";
    }
    enum "locally-configured" {
      description "Configured entity.";
    }
    enum "ospfv2" {
      description "OSPFv2.";
    }
    enum "ospfv3" {
      description "OSPFv3.";
    }
    enum "isis" {
      description "ISIS.";
    }
    enum "bgp-ls" {
      description "BGP-LS.";
      reference "RFC 7752: North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP";
    }
    enum "system-processed" {
      description "System processed entity.";
    }
    enum "other" {
      description "Other source.";
    }
  }
  description
    "Describing the type of source that has provided the related information, and the source credibility.";
} // te-info-source

#if 0
/*
 * Groupings
 */
#endif

grouping connectivity-matrix-entry-path-attributes {
  description
    "...";
}
leaf is-allowed {
  type boolean;
  description
    "true - switching is allowed,
    false - switching is disallowed.";
}

container underlay {
  if-feature te-topology-hierarchy;
  description "Attributes of the te-link underlay.";
  reference
    "RFC 4206: Label Switched Paths (LSP) Hierarchy with
     Generalized Multi-Protocol Label Switching (GMPLS)
     Traffic Engineering (TE)";

  uses te-link-underlay-attributes;
} // underlay

uses te-types:generic-path-constraints;
uses te-types:generic-path-optimization;
uses te-types:generic-path-properties;
} // connectivity-matrix-entry-path-attributes

grouping geolocation-container {
  description
    "A container containing a GPS location.";
  container geolocation{
    config false;
    description
      "A container containing a GPS location.";
    leaf altitude {
      type int64;
      units millimeter;
      description
        "Distance above the sea level.";
    }
    leaf latitude {
      type geographic-coordinate-degree {
        range "-90..90";
      }
      description
    }
  }
}
"Relative position north or south on the Earth’s surface.";
}
leaf longitude {
  type geographic-coordinate-degree {
    range "-180..180";
  }
  description
  "Angular distance east or west on the Earth’s surface.";
}
} // gps-location
} // geolocation-container

grouping information-source-state-attributes {
  description
  "The attributes identifying source that has provided the related information, and the source credibility.";
  leaf credibility-preference {
    type uint16;
    description
    "The preference value to calculate the traffic engineering database credibility value used for tie-break selection between different information-source values.
Higher value is more preferable.";
  }
  leaf logical-network-element {
    type string;
    description
    "When applicable, this is the name of a logical network element from which the information is learned.";
  } // logical-network-element
  leaf network-instance {
    type string;
    description
    "When applicable, this is the name of a network-instance from which the information is learned.";
  } // network-instance
} // information-source-state-attributes

grouping information-source-per-link-attributes {
  description
"Per node container of the attributes identifying source that has provided the related information, and the source credibility.";
leaf information-source {
  type te-info-source;
  config false;
  description
  "Indicates the type of the information source.";
}
leaf information-source-instance {
  type string;
  config false;
  description
  "The name indicating the instance of the information source.";
}
container information-source-state {
  config false;
  description
  "The container contains state attributes related to the information source.";
  uses information-source-state-attributes;
  container topology {
    description
    "When the information is processed by the system, the attributes in this container indicate which topology is used to process to generate the result information.";
    uses nt:link-ref;
  } // topology
} // information-source-state
} // information-source-per-link-attributes

grouping information-source-per-node-attributes {
  description
  "Per node container of the attributes identifying source that has provided the related information, and the source credibility.";
  leaf information-source {
    type te-info-source;
    config false;
    description

"Indicates the type of the information source."
}
leaf information-source-instance {
  type string;
  config false;
  description
    "The name indicating the instance of the information source.";
}
container information-source-state {
  config false;
  description
    "The container contains state attributes related to the information source.";
  uses information-source-state-attributes;
  container topology {
    description
      "When the information is processed by the system, the attributes in this container indicate which topology is used to process to generate the result information.";
    uses nw:node-ref;
  } // topology
} // information-source-state
} // information-source-per-node-attributes

grouping interface-switching-capability-list {
  description
    "List of Interface Switching Capabilities Descriptors (ISCD)";
  list interface-switching-capability {
    key "switching-capability encoding";
    description
      "List of Interface Switching Capabilities Descriptors (ISCD) for this link.";
    reference
      "RFC 3471: Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description."
      "RFC 4203: OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)."
    leaf switching-capability {
      type identityref {
        base te-types:switching-capabilities;
description
  "Switching Capability for this interface."
}
leaf encoding {
  type identityref {
    base te-types:lsp-encoding-types;
  }
  description
    "Encoding supported by this interface.";
}
  uses te-link-iscd-attributes;
} // interface-switching-capability
} // interface-switching-capability-list

grouping statistics-per-link {
  description
    "Statistics attributes per TE link.";
  leaf discontinuity-time {
    type yang:date-and-time;
    description
      "The time on the most recent occasion at which any one or
      more of this interface’s counters suffered a
discontinuity. If no such discontinuities have occurred
since the last re-initialization of the local management
subsystem, then this node contains the time the local
management subsystem re-initialized itself.";
  }
  /* Administrative attributes */
  leaf disables {
    type yang:counter32;
    description
      "Number of times that link was disabled.";
  }
  leaf enables {
    type yang:counter32;
    description
      "Number of times that link was enabled.";
  }
  leaf maintenance-clears {
    type yang:counter32;
  }

description
  "Number of times that link was put out of maintenance.";
}
leaf maintenance-sets {
  type yang:counter32;
  description
  "Number of times that link was put in maintenance.";
}
leaf modifies {
  type yang:counter32;
  description
  "Number of times that link was modified.";
}
/* Operational attributes */
leaf downs {
  type yang:counter32;
  description
  "Number of times that link was set to operational down.";
}
leaf ups {
  type yang:counter32;
  description
  "Number of times that link was set to operational up.";
}
/* Recovery attributes */
leaf fault-clears {
  type yang:counter32;
  description
  "Number of times that link experienced fault clear event.";
}
leaf fault-detects {
  type yang:counter32;
  description
  "Number of times that link experienced fault detection.";
}
leaf protection-switches {
  type yang:counter32;
  description
  "Number of times that link experienced protection switchover.";
}
leaf protection-reverts {
  type yang:counter32;
  description
    "Number of times that link experienced protection
    reversion.";
}
leaf restoration-failures {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    failure.";
}
leaf restoration-starts {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    start.";
}
leaf restoration-successes {
  type yang:counter32;
  description
    "Number of times that link experienced restoration
    success.";
}
leaf restoration-reversion-failures {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    failure.";
}
leaf restoration-reversion-starts {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    start.";
}
leaf restoration-reversion-successes {
  type yang:counter32;
  description
    "Number of times that link experienced restoration reversion
    success.";
}
grouping {  
  description "Statistics attributes per TE node.";  
  leaf discontinuity-time {    
    type yang:date-and-time;    
    description "The time on the most recent occasion at which any one or more of this interface’s counters suffered a discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem re-initialized itself.";  
  }  
}  

container {  
  description "Containing TE node level statistics attributes.";  
  leaf disables {   
    type yang:counter32;   
    description "Number of times that node was disabled.";  
  }  
  leaf enables {    
    type yang:counter32;    
    description "Number of times that node was enabled.";  
  }  
  leaf maintenance-sets {   
    type yang:counter32;   
    description "Number of times that node was put in maintenance.";  
  }  
  leaf maintenance-clears {   
    type yang:counter32;   
    description "Number of times that node was put out of maintenance.";  
  }  
  leaf modifies {   
    type yang:counter32;   
  }
container connectivity-matrix-entry {
    description
    "Containing connectivity matrix entry level statistics attributes.";
    leaf creates {
        type yang:counter32;
        description
        "Number of times that a connectivity matrix entry was created.";
        reference
        "RFC 6241. Section 7.2 for 'create' operation.";
    }
    leaf deletes {
        type yang:counter32;
        description
        "Number of times that a connectivity matrix entry was deleted.";
        reference
        "RFC 6241. Section 7.2 for 'delete' operation.";
    }
    leaf disables {
        type yang:counter32;
        description
        "Number of times that a connectivity matrix entry was disabled.";
    }
    leaf enables {
        type yang:counter32;
        description
        "Number of times that a connectivity matrix entry was enabled.";
    }
    leaf modifies {
        type yang:counter32;
        description
        "Number of times that a connectivity matrix entry was modified.";
    }
}
grouping statistics-per-ttp {
  description "Statistics attributes per TE TTP (Tunnel Termination Point).";
  leaf discontinuity-time {
    type yang:date-and-time;
    description "The time on the most recent occasion at which any one or
    more of this interface’s counters suffered a discontinuity. If no such
    discontinuities have occurred since the last re-initialization of the local management
    subsystem, then this node contains the time the local management subsystem re-initialized itself.";
  }
}

container tunnel-termination-point {
  description "Containing TE TTP (Tunnel Termination Point) level
  statistics attributes.";
  /* Administrative attributes */
  leaf disables {
    type yang:counter32;
    description "Number of times that TTP was disabled.";
  }
  leaf enables {
    type yang:counter32;
    description "Number of times that TTP was enabled.";
  }
  leaf maintenance-clears {
    type yang:counter32;
    description "Number of times that TTP was put out of maintenance.";
  }
  leaf maintenance-sets {
    type yang:counter32;
    description "Number of times that TTP was put in maintenance.";
  }
}
leaf modifies {
    type yang:counter32;
    description
    "Number of times that TTP was modified.";
}

/* Operational attributes */
leaf downs {
    type yang:counter32;
    description
    "Number of times that TTP was set to operational down.";
}
leaf ups {
    type yang:counter32;
    description
    "Number of times that TTP was set to operational up.";
}
leaf in-service-clears {
    type yang:counter32;
    description
    "Number of times that TTP was taken out of service
    (TE tunnel was released).";
}
leaf in-service-sets {
    type yang:counter32;
    description
    "Number of times that TTP was put in service by a TE
    tunnel (TE tunnel was set up).";
}

} // tunnel-termination-point

container local-link-connectivity {
    description
    "Containing TE LLCL (Local Link Connectivity List) level
    statistics attributes.";
    leaf creates {
        type yang:counter32;
        description
        "Number of times that an LLCL entry was created.";
        reference
        "RFC 6241. Section 7.2 for 'create' operation.";
    }
}
leaf deletes {
    type yang:counter32;
    description "Number of times that an LLCL entry was deleted."
    reference "RFC 6241. Section 7.2 for 'delete' operation.";
}
leaf disables {
    type yang:counter32;
    description "Number of times that an LLCL entry was disabled."
}
leaf enables {
    type yang:counter32;
    description "Number of times that an LLCL entry was enabled."
}
leaf modifies {
    type yang:counter32;
    description "Number of times that an LLCL entry was modified."
}
} // local-link-connectivity
} // statistics-per-ttp

grouping te-link-augment {
    description "Augmentation for TE link.";
    uses te-link-config;
    uses te-link-state-derived;
    container statistics {
        config false;
        description "Statistics data."
        uses statistics-per-link;
    } // statistics
} // te-link-augment

grouping te-link-config {
    description
"TE link configuration grouping.";
choice bundle-stack-level {
  description
  "The TE link can be partitioned into bundled
  links, or component links.";
  case bundle {
    container bundled-links {
      description
      "A set of bundled links.";
      reference
      "RFC 4201: Link Bundling in MPLS Traffic Engineering
      (TE).";
      list bundled-link {
        key "sequence";
        description
        "Specify a bundled interface that is
        further partitioned.";
        leaf sequence {
          type uint32;
          description
          "Identify the sequence in the bundle.";
        }
      } // list bundled-link
    }
  }
  case component {
    container component-links {
      description
      "A set of component links"
      list component-link {
        key "sequence";
        description
        "Specify a component interface that is
        sufficient to unambiguously identify the
        appropriate resources";
        leaf sequence {
          type uint32;
          description
          "Identify the sequence in the bundle.";
        }
      }
    }
  }
} // choice bundle-stack-level
leaf src-interface-ref {
   type string;
   description
   "Reference to component link interface on the
   source node.";
}
leaf des-interface-ref {
   type string;
   description
   "Reference to component link interface on the
   destination node.";
}

leaf-list te-link-template {
   if-feature template;
   type leafref {
      path "../../../..//te/templates/link-template/name";
   } 
   description
   "The reference to a TE link template.";
} 
uses te-link-config-attributes;
} // te-link-config

grouping te-link-config-attributes {
   description
   "Link configuration attributes in a TE topology.";
container te-link-attributes {
   description "Link attributes in a TE topology.";
   leaf access-type {
      type te-types:te-link-access-type;
      description
      "Link access type, which can be point-to-point or
      multi-access.";
   }
container external-domain {
   description
   "";
}
"For an inter-domain link, specify the attributes of the remote end of link, to facilitate the signalling at local end."

uses nw:network-ref;

leaf remote-te-node-id {
  type te-types:te-node-id;
  description "Remote TE node identifier, used together with remote-te-link-id to identify the remote link termination point in a different domain.";
}

leaf remote-te-link-tp-id {
  type te-types:te-tp-id;
  description "Remote TE link termination point identifier, used together with remote-te-node-id to identify the remote link termination point in a different domain.";
}

leaf is-abstract {
  type empty;
  description "Present if the link is abstract.";
}

leaf name {
  type string;
  description "Link Name.";
}

container underlay {
  if-feature te-topology-hierarchy;
  description "Attributes of the te-link underlay.";

  uses te-link-underlay-attributes;
} // underlay

leaf admin-status {
  type te-types:te-admin-status;
  description "The administrative state of the link.";
}
uses te-link-info-attributes;
} // te-link-attributes
} // te-link-config-attributes

grouping te-link-info-attributes {
  description
    "Advertised TE information attributes.";
  leaf link-index {
    type uint64;
    description
      "The link identifier. If OSPF is used, this represents an
      ospfLsdbID. If IS-IS is used, this represents an isisLSPID.
      If a locally configured link is used, this object represents
      a unique value, which is locally defined in a router.";
  }
  leaf administrative-group {
    type te-types:admin-groups;
    description
      "Administrative group or color of the link.
      This attribute covers both administrative group (defined in
      RFC 3630, RFC 5305 and RFC 5329), and extended
      administrative group (defined in RFC 7308).";
  }
} uses interface-switching-capability-list;
uses te-types:label-set-info;

leaf link-protection-type {
  type identityref {
    base te-types:link-protection-type;
  }
  description
    "Link Protection Type desired for this link.";
  reference
    "RFC 4202: Routing Extensions in Support of
    Generalized Multi-Protocol Label Switching (GMPLS).";
}

container max-link-bandwidth {

uses te-types:te-bandwidth;
description
"Maximum bandwidth that can be seen on this link in this
direction. Units in bytes per second.";
reference
"RFC 3630: Traffic Engineering (TE) Extensions to OSPF
Version 2.  
RFC 5305: IS-IS Extensions for Traffic Engineering.";
}
container max-resv-link-bandwidth {
uses te-types:te-bandwidth;
description
"Maximum amount of bandwidth that can be reserved in this
direction in this link. Units in bytes per second.";
reference
"RFC 3630: Traffic Engineering (TE) Extensions to OSPF
Version 2.  
RFC 5305: IS-IS Extensions for Traffic Engineering.";
}
list unreserved-bandwidth {
  key "priority";
  max-elements "8";
  description
  "Unreserved bandwidth for 0-7 priority levels. Units in
  bytes per second.";
  reference
  "RFC 3630: Traffic Engineering (TE) Extensions to OSPF
  Version 2.  
  RFC 5305: IS-IS Extensions for Traffic Engineering.";
  leaf priority {
    type uint8 {
      range "0..7";
    }
    description "Priority.";
  }
  uses te-types:te-bandwidth;
}
leaf te-default-metric {
  type uint32;
  description
  "Traffic engineering metric.";
reference
"RFC 3630: Traffic Engineering (TE) Extensions to OSPF
Version 2.
RFC 5305: IS-IS Extensions for Traffic Engineering.";
leaf te-delay-metric {
type uint32;
description
"Traffic engineering delay metric."
reference
"RFC 7471: OSPF Traffic Engineering (TE) Metric Extensions.";
}
leaf te-igp-metric {
type uint32;
description
"IGP metric used for traffic engineering."
reference
"RFC 3785: Use of Interior Gateway Protocol (IGP) Metric as a
Second MPLS Traffic Engineering (TE) Metric.";
}
container te-srlgs {
description
"Containing a list of SRLGs.";
leaf-list value {
type te-types:srlg;
description "SRLG value."
reference
"RFC 4202: Routing Extensions in Support of
Generalized Multi-Protocol Label Switching (GMPLS).";
}
}
container te-nsrlgs {
if-feature nsrlg;
description
"Containing a list of NSRLGs (Not Sharing Risk Link Groups).
When an abstract TE link is configured, this list specifies
the request that underlay TE paths need to be mutually
disjoint with other TE links in the same groups.";
leaf-list id {
type uint32;
}
description
"NSRLG ID, uniquely configured within a topology."
reference
"RFC 4872: RSVP-TE Extensions in Support of End-to-End
Generalized Multi-Protocol Label Switching (GMPLS)
Recovery";
}
} // te-link-info-attributes

grouping te-link-iscd-attributes {
  description
  "TE link ISCD (Interface Switching Capability Descriptor)
  attributes.";
  reference
  "Sec 1.4, RFC 4203: OSPF Extensions in Support of Generalized
  Multi-Protocol Label Switching (GMPLS). Section 1.4."
  list max-lsp-bandwidth {
    key "priority";
    max-elements "8"
    description
    "Maximum LSP Bandwidth at priorities 0-7."
    leaf priority {
      type uint8 {
        range "0..7";
      }
      description "Priority."
    } // te-types:te-bandwidth
  } // te-link-iscd-attributes

grouping te-link-state-derived {
  description
  "Link state attributes in a TE topology.";
  leaf oper-status {
    type te-types:te-oper-status;
    config false;
    description
    "The current operational state of the link.";
  }
} // te-link-state-derived

Liu, et al Expires December 19, 2019 [Page 68]
leaf is-transitional {
  type empty;
  config false;
  description
    "Present if the link is transitional, used as an
    alternative approach in lieu of inter-layer-lock-id
    for path computation in a TE topology covering multiple
    layers or multiple regions."
  reference
    "RFC 5212: Requirements for GMPLS-Based Multi-Region and
    Multi-Layer Networks (MRN/MLN).
    RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
    for Multi-Layer and Multi-Region Networks (MLN/MRN)."
}
uses information-source-per-link-attributes;
list information-source-entry {
  key "information-source information-source-instance";
  config false;
  description
    "A list of information sources learned, including the one
    used.";
  uses information-source-per-link-attributes;
  uses te-link-info-attributes;
}
container recovery {
  config false;
  description
    "Status of the recovery process.";
  leaf restoration-status {
    type te-types:te-recovery-status;
    description
      "Restoration status.";
  }
  leaf protection-status {
    type te-types:te-recovery-status;
    description
      "Protection status.";
  }
}
container underlay {
  if-feature te-topology-hierarchy;
config false;
description "State attributes for te-link underlay."
leaf dynamic {
type boolean;
description
"true if the underlay is dynamically created.";
}
leaf committed {
type boolean;
description
"true if the underlay is committed.";
}
} // te-link-state-derived

grouping te-link-underlay-attributes {
description "Attributes for te-link underlay.";
reference
"RFC 4206: Label Switched Paths (LSP) Hierarchy with
Generalized Multi-Protocol Label Switching (GMPLS)
Traffic Engineering (TE)";
leaf enabled {
type boolean;
description
"'true' if the underlay is enabled.
'false' if the underlay is disabled.";
}
container primary-path {
description
"The service path on the underlay topology that
supports this link.";
uses nw:network-ref;
list path-element {
key "path-element-id";
description
"A list of path elements describing the service path.";
leaf path-element-id {
type uint32;
description "To identify the element in a path.";
}
uses te-path-element;
}
list backup-path {
  key "index";
  description
  "A list of backup service paths on the underlay topology that
  protect the underlay primary path. If the primary path is
  not protected, the list contains zero elements. If the
  primary path is protected, the list contains one or more
  elements.";
  leaf index {
    type uint32;
    description
    "A sequence number to identify a backup path.";
  }
  uses nw:network-ref;
  list path-element {
    key "path-element-id";
    description
    "A list of path elements describing the backup service
    path";
    leaf path-element-id {
      type uint32;
      description "To identify the element in a path.";
    }
    uses te-path-element;
  }
  // underlay-backup-path
  leaf protection-type {
    type identityref {
      base te-types:lsp-protection-type;
    }
    description
    "Underlay protection type desired for this link.";
  }
  container tunnel-termination-points {
    description
    "Underlay TTP(Tunnel Termination Points) desired for this
    link.";
    leaf source {
      type binary;
    }
  }
} // primary-path
leaf destination {
  type binary;
  description
  "Destination tunnel termination point identifier.";
}

container tunnels {
  description
  "Underlay TE tunnels supporting this TE link.";
  leaf sharing {
    type boolean;
    default true;
    description
    "'true' if the underlay tunnel can be shared with other
    TE links;
    'false' if the underlay tunnel is dedicated to this
    TE link.
    This leaf is the default option for all TE tunnels,
    and may be overridden by the per TE tunnel value.";
  }
  
  list tunnel {
    key "tunnel-name";
    description
    "Zero, one or more underlay TE tunnels that support this TE
    link.";
    leaf tunnel-name {
      type string;
      description
      "A tunnel name uniquely identifies an underlay TE tunnel,
      used together with the source-node of this link.
      The detailed information of this tunnel can be retrieved
      from the ietf-te model.";
      reference "RFC 3209";
    }
    leaf sharing {
      type boolean;
      description
      "'true' if the underlay tunnel can be shared with other
      
      Liu, et al Expires December 19, 2019 [Page 72]
TE links;
'false' if the underlay tunnel is dedicated to this
TE link.
}
} // tunnel
} // tunnels
} // te-link-underlay-attributes

grouping te-node-augment {
  description
    "Augmentation for TE node.";
  uses te-node-config;
  uses te-node-state-derived;
  container statistics {
    config false;
    description
      "Statistics data.";
    uses statistics-per-node;
  } // statistics

  list tunnel-termination-point {
    key "tunnel-tp-id";
    description
      "A termination point can terminate a tunnel.";
    leaf tunnel-tp-id {
      type binary;
      description
        "Tunnel termination point identifier.";
    }
  }

  uses te-node-tunnel-termination-point-config;
  leaf oper-status {
    type te-types:te-oper-status;
    config false;
    description
      "The current operational state of the tunnel
       termination point.";
  }

  uses geolocation-container;
  container statistics {
    config false;
description
  "Statistics data."
uses statistics-per-ttp;
} // statistics

// Relations to other tunnel termination points
list supporting-tunnel-termination-point {
  key "node-ref tunnel-tp-ref";
  description
  "Identifies the tunnel termination points, that this
tunnel termination point is depending on.";
  leaf node-ref {
    type inet:uri;
    description
    "This leaf identifies the node in which the supporting
tunnel termination point is present.
This node is either the supporting node or a node in
an underlay topology.";
  }
  leaf tunnel-tp-ref {
    type binary;
    description
    "Reference to a tunnel termination point, which is
either in the supporting node or a node in an
underlay topology.";
  }
} // supporting-tunnel-termination-point
} // tunnel-termination-point
} // te-node-augment

grouping te-node-config {
  description "TE node configuration grouping.";
  leaf-list te-node-template {
    if-feature template;
    type leafref {
      path "../../../../te/templates/node-template/name";
    }
    description
    "The reference to a TE node template.";
  }
  uses te-node-config-attributes;
} // te-node-config

Liu, et al Expires December 19, 2019 [Page 74]
grouping te-node-config-attributes {
  description "Configuration node attributes in a TE topology.";
  container te-node-attributes {
    description "Containing node attributes in a TE topology.";
    leaf admin-status {
      type te-types:te-admin-status;
      description
        "The administrative state of the link.";
    }
    uses te-node-connectivity-matrices;
    uses te-node-info-attributes;
  }
}

} // te-node-config-attributes

grouping te-node-config-attributes-template {
  description
    "Configuration node attributes for template in a TE topology.";
  container te-node-attributes {
    description "Containing node attributes in a TE topology.";
    leaf admin-status {
      type te-types:te-admin-status;
      description
        "The administrative state of the link.";
    }
    uses te-node-info-attributes;
  }
}

} // te-node-config-attributes-template

grouping te-node-connectivity-matrices {
  description "Connectivity matrix on a TE node.";
  container connectivity-matrices {
    description
      "Containing connectivity matrix on a TE node.";
    leaf number-of-entries {
      type uint16;
      description
        "The number of connectivity matrix entries.
         If this number is specified in the configuration request, 
         the number is requested number of entries, which may not
all be listed in the list;
if this number is reported in the state data,
the number is the current number of operational entries.";
}
uses te-types:label-set-info;
uses connectivity-matrix-entry-path-attributes;
list connectivity-matrix {
  key "id";
  description
    "Represents node’s switching limitations, i.e. limitations
    in interconnecting network TE links across the node.";
  reference
    "RFC 7579: General Network Element Constraint Encoding
    for GMPLS-Controlled Networks.";
  leaf id {
    type uint32;
    description "Identifies the connectivity-matrix entry.";
  }
} // connectivity-matrix
} // connectivity-matrices
} // te-node-connectivity-matrices

grouping te-node-connectivity-matrix-attributes {
  description
    "Termination point references of a connectivity matrix entry.";
  container from {
    description
      "Reference to source link termination point.";
    leaf tp-ref {
      type leafref {
        path "../../../../../nt:termination-point/nt:tp-id";
      }
      description
        "Relative reference to a termination point.";
    }
    uses te-types:label-set-info;
  }
  container to {
    description
      "Reference to destination link termination point.";
    leaf tp-ref {

type leafref {
    path "../../../../../nt:termination-point/nt:tp-id";
} description
    "Relative reference to a termination point."
} uses te-types:label-set-info;
} uses connectivity-matrix-entry-path-attributes;
} // te-node-connectivity-matrix-attributes

grouping te-node-info-attributes {
    description
        "Advertised TE information attributes.";
    leaf domain-id {
        type uint32;
        description
            "Identifies the domain that this node belongs. This attribute is used to support inter-domain links.";
        reference
            "RFC 5152: A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs).
            RFC 5316: ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering."
    } leaf is-abstract {
        type empty;
        description
            "Present if the node is abstract, not present if the node is actual.";
    } leaf name {
        type string;
        description "Node name.";
    } leaf-list signaling-address {
        type inet:ip-address;
        description "Node signaling address.";
container underlay-topology {
  if-feature te-topology-hierarchy;
  description
    "When an abstract node encapsulates a topology, 
     the attributes in this container point to said topology.";
  uses nw:network-ref;
}
} // te-node-info-attributes

grouping te-node-state-derived {
  description "Node state attributes in a TE topology.";
  leaf oper-status {
    type te-types:te-oper-status;
    config false;
    description
      "The current operational state of the node.";
    uses geolocation-container;
  }
  leaf is-multi-access-dr {
    type empty;
    config false;
    description
      "The presence of this attribute indicates that this TE node 
       is a pseudonode elected as a designated router.";
    reference
      "RFC 3630: Traffic Engineering (TE) Extensions to OSPF 
       Version 2.
      RFC 1195: Use of OSI IS-IS for Routing in TCP/IP and Dual 
       Environments.";
  }
  uses information-source-per-node-attributes;
  list information-source-entry {
    key "information-source information-source-instance";
    config false;
    description
      "A list of information sources learned, including the one 
       used.";
    uses information-source-per-node-attributes;
    uses te-node-connectivity-matrices;
    uses te-node-info-attributes;
  }


grouping te-node-tunnel-termination-point-config {
    description
    "Termination capability of a tunnel termination point on a
    TE node.";
    uses te-node-tunnel-termination-point-config-attributes;
    container local-link-connectivities {
        description
        "Containing local link connectivity list for
        a tunnel termination point on a TE node.";
        leaf number-of-entries {
            type uint16;
            description
            "The number of local link connectivity list entries.
            If this number is specified in the configuration request,
            the number is requested number of entries, which may not
            all be listed in the list;
            if this number is reported in the state data,
            the number is the current number of operational entries.";
        }
        uses te-types:label-set-info;
        uses connectivity-matrix-entry-path-attributes;
    } // local-link-connectivities
} // te-node-tunnel-termination-point-config

grouping te-node-tunnel-termination-point-config-attributes {
    description
    "Configuration attributes of a tunnel termination point on a
    TE node.";
    leaf admin-status {
        type te-types:te-admin-status;
        description
        "The administrative state of the tunnel termination point.";
    }
    leaf name {
        type string;
        description
        "A descriptive name for the tunnel termination point.";
    }
}
leaf switching-capability {
  type identityref {
    base te-types:switching-capabilities;
  }
  description
    "Switching Capability for this interface.";
}
leaf encoding {
  type identityref {
    base te-types:lsp-encoding-types;
  }
  description
    "Encoding supported by this interface.";
}
leaf-list inter-layer-lock-id {
  type uint32;
  description
    "Inter layer lock ID, used for path computation in a TE topology covering multiple layers or multiple regions.";
  reference
    "RFC 5212: Requirements for GMPLS-Based Multi-Region and Multi-Layer Networks (MRN/MLN).
RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions for Multi-Layer and Multi-Region Networks (MLN/MRN).";
}
leaf protection-type {
  type identityref {
    base te-types:lsp-protection-type;
  }
  description
    "The protection type that this tunnel termination point is capable of.";
}
ccontainer client-layer-adaptation {
  description
    "Containing capability information to support a client layer adaptation in multi-layer topology.";
  list switching-capability {
    key "switching-capability encoding";
    description
  }
}
"List of supported switching capabilities";
reference
RFC 4202: Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS).";
leaf switching-capability {
  type identityref {
    base te-types:switching-capabilities;
  }
  description
    "Switching Capability for the client layer adaption.";
}
leaf encoding {
  type identityref {
    base te-types:lsp-encoding-types;
  }
  description
    "Encoding supported by the client layer adaption.";
}
uses te-types:te-bandwidth;
}
} // te-node-tunnel-termination-point-config-attributes

grouping te-node-tunnel-termination-point-llc-list {
  description
    "Local link connectivity list of a tunnel termination point on a TE node.";
  list local-link-connectivity {
    key "link-tp-ref";
    description
      "The termination capabilities between tunnel-termination-point and link termination-point.
The capability information can be used to compute the tunnel path.
The Interface Adjustment Capability Descriptors (IACD) (defined in RFC 6001) on each link-tp can be derived from this local-link-connectivity list.";
    reference
      "RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions...";
  }
}

Liu, et al Expires December 19, 2019 [Page 81]
leaf link-tp-ref {
  type leafref {
    path "../../../nt:termination-point/nt:tp-id";
  }
  description
  "Link termination point.";
}
uses te-types:label-set-info;
uses connectivity-matrix-entry-path-attributes;
} // local-link-connectivity
} // te-node-tunnel-termination-point-config

grouping te-path-element {
  description
  "A group of attributes defining an element in a TE path such as TE
node, TE link, TE atomic resource or label.";
  uses te-types:explicit-route-hop;
} // te-path-element

grouping te-termination-point-augment {
  description
  "Augmentation for TE termination point.";
  leaf te-tp-id {
    type te-types:te-tp-id;
    description
    "An identifier to uniquely identify a TE termination point.";
  }
  container te {
    must "../te-tp-id";
    presence "TE support.";
    description
    "Indicates TE support.";
    uses te-termination-point-config;
    leaf oper-status {
      type te-types:te-oper-status;
      config false;
      description
    }
  }
} // te-termination-point-augment
"The current operational state of the link termination point."
}
uses geolocation-container;
} // te
} // te-termination-point-augment

grouping te-termination-point-config {
  description
  "TE termination point configuration grouping.";
  leaf admin-status {
    type te-types:te-admin-status;
    description
    "The administrative state of the link termination point.";
  }
  leaf name {
    type string;
    description
    "A descriptive name for the link termination point.";
  }
  uses interface-switching-capability-list;
  leaf inter-domain-plug-id {
    type binary;
    description
    "A topology-wide unique number that identifies on the network a connectivity supporting a given inter-domain TE link. This is more flexible alternative to specifying remote-te-node-id and remote-te-link-tp-id on a TE link, when the provider does not know remote-te-node-id and remote-te-link-tp-id or need to give client the flexibility to mix-n-match multiple topologies.";
  }
  leaf-list inter-layer-lock-id {
    type uint32;
    description
    "Inter layer lock ID, used for path computation in a TE topology covering multiple layers or multiple regions.";
    reference
    "RFC 5212: Requirements for GMPLS-Based Multi-Region and Multi-Layer Networks (MRN/MLN).
    RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions
    
    Liu, et al Expires December 19, 2019 [Page 83]"
for Multi-Layer and Multi-Region Networks (MLN/MRN).";

} // te-termination-point-config

grouping te-topologies-augment {
    description "Augmentation for TE topologies.";
    container te {
        presence "TE support.";
        description "Indicates TE support.";

        container templates {
            description "Configuration parameters for templates used for TE topology.";

            list node-template {
                if-feature template;
                key "name";
                leaf name {
                    type te-types:te-template-name;
                    description "The name to identify a TE node template.";
                }
                description "The list of TE node templates used to define sharable and reusable TE node attributes.";
                uses template-attributes;
                uses te-node-config-attributes-template;
            } // node-template

            list link-template {
                if-feature template;
                key "name";
                leaf name {
                    type te-types:te-template-name;
                    description "The name to identify a TE link template.";
                }
                description
            } // link-template
        }
    }
}

Liu, et al Expires December 19, 2019 [Page 84]
"The list of TE link templates used to define sharable and reusable TE link attributes."
uses template-attributes;
uses te-link-config-attributes;
} // link-template
} // templates
} // te
} // te-topologies-augment

grouping te-topology-augment {
  description
  "Augmentation for TE topology.";
  uses te-types:te-topology-identifier;

ccontainer te {
  must ".../te-topology-identifier/provider-id"
  + " and ../te-topology-identifier/client-id"
  + " and ../te-topology-identifier/topology-id";
  presence "TE support.";
  description
  "Indicates TE support.";

  uses te-topology-config;
  uses geolocation-container;
} // te
} // te-topology-augment


grouping te-topology-config {
  description
  "TE topology configuration grouping.";
  leaf name {
    type string;
    description
    "Name of the TE topology. This attribute is optional and can be specified by the operator to describe the TE topology, which can be useful when network-id is not descriptive and not modifiable because of being generated by the system.";
  }
  leaf preference {
    type uint8 {

range "1..255";
}
description
"Specifies a preference for this topology. A lower number indicates a higher preference."
}
leaf optimization-criterion {
type identityref {
  base te-types:objective-function-type;
}
description
"Optimization criterion applied to this topology."
reference
"RFC 3272: Overview and Principles of Internet Traffic Engineering."
}
list nsrlg {
  if-feature nsrlg;
  key "id";
  description
  "List of NSRLGs (Not Sharing Risk Link Groups)."
  reference
  "RFC 4872: RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery";
  leaf id {
    type uint32;
    description
    "Identify the NSRLG entry."
  }
  leaf disjointness {
    type te-types:te-path-disjointness;
    description
    "The type of resource disjointness."
  }
} // nsrlg
} // te-topology-config

grouping template-attributes {
  description
  "Common attributes for all templates."
}
leaf priority {
  type uint16;
  description
      "The preference value to resolve conflicts between different
      templates. When two or more templates specify values for
      one configuration attribute, the value from the template
      with the highest priority is used.
      A lower number indicates a higher priority. The highest
      priority is 0.";
}

leaf reference-change-policy {
  type enumeration {
    enum no-action {
      description
          "When an attribute changes in this template, the
          configuration node referring to this template does
          not take any action.";
    }
    enum not-allowed {
      description
          "When any configuration object has a reference to this
          template, changing this template is not allowed.";
    }
    enum cascade {
      description
          "When an attribute changes in this template, the
          configuration object referring to this template applies
          the new attribute value to the corresponding
          configuration.";
    }
  }
  description
      "This attribute specifies the action taken to a configuration
      node that has a reference to this template.";
}

/*
 * Data nodes
 */
augment "/nw:networks/nw:network/nw:network-types" {
augment "/nw:networks/nw:network" {
    when "nw:network-types/tet:te-topology" {
        description
        "Augmentation parameters apply only for networks with TE topology type.";
    }
    description
    "Configuration parameters for TE topology.";
    uses te-topology-augment;
}

augment "/nw:networks/nw:network/nw:node" {
    when ".../nw:network-types/tet:te-topology" {
        description
        "Augmentation parameters apply only for networks with TE topology type.";
    }
    description
    "Configuration parameters for TE at node level.";
    leaf te-node-id {
        type te-types:te-node-id;
        description
        "The identifier of a node in the TE topology.
        A node is specific to a topology to which it belongs.";
    }
    container te {
    }
must "./te-node-id" {
  description
  "te-node-id is mandatory.";
}
must "./nw:supporting-node)<=1" {
  description
  "For a node in a TE topology, there cannot be more
  than 1 supporting node. If multiple nodes are abstracted,
  the underlay-topology is used.";
} presence "TE support.";
  description
  "Indicates TE support.";
  uses te-node-augment;
} // te

augment "/nw:networks/nw:network/nt:link" {
  when "./nw:network-types/tet:te-topology" {
    description
    "Augmentation parameters apply only for networks with
    TE topology type.";
  }
  description
  "Configuration parameters for TE at link level.";
  container te {
    must "./nt:supporting-link)<=1" {
      description
      "For a link in a TE topology, there cannot be more
      than 1 supporting link. If one or more link paths are
      abstracted, the underlay is used.";
    }
    presence "TE support.";
    description
    "Indicates TE support.";
    uses te-link-augment;
  } // te
}

augment "/nw:networks/nw:network/nw:node/" 
+ "nt:termination-point" {
when "../nw:network-types/tet:te-topology" {  
  description  
  "Augmentation parameters apply only for networks with  
  TE topology type.";
}

augment 
  "nw:networks/nw:network/nt:link/te/bundle-stack-level/"  
  + "bundle/bundled-links/bundled-link" {  
    when "../nw:network-types/tet:te-topology" {  
      description  
      "Augmentation parameters apply only for networks with  
      TE topology type.";
    }
    description  
    "Augment TE link bundled link.";
    leaf src-tp-ref {  
      type leafref {  
        path "..//nw:node[nw:node-id =  
          + "current()//nt:source/"  
          + "nt:source-node]//  
          + nt:termination-point/nt:tp-id";
        require-instance true;
      }
      description  
      "Reference to another TE termination point on the  
      same source node.";
    }
    leaf des-tp-ref {  
      type leafref {  
        path "..//nw:node[nw:node-id =  
          + "current()//nt:destination/"  
          + "nt:dest-node]//  
          + nt:termination-point/nt:tp-id";
        require-instance true;
      }
      description
  }
"Reference to another TE termination point on the same destination node.";
}

augment
  /nw:networks/nw:network/nw:node/te/
  + "information-source-entry/connectivity-matrices/"
  + "connectivity-matrix"
  when "../../../nw:network-types/tet:te-topology"
  description
    "Augmentation parameters apply only for networks with TE topology type.";
}

description
  "Augment TE node connectivity-matrix.";
uses te-node-connectivity-matrix-attributes;
}

augment
  /nw:networks/nw:network/nw:node/te/te-node-attributes/
  + "connectivity-matrices/connectivity-matrix"
  when "../../../nw:network-types/tet:te-topology"
  description
    "Augmentation parameters apply only for networks with TE topology type.";
}

description
  "Augment TE node connectivity-matrix.";
uses te-node-connectivity-matrix-attributes;
}

augment
  /nw:networks/nw:network/nw:node/te/
  + "tunnel-termination-point/local-link-connectivities/"
  when "../../../nw:network-types/tet:te-topology"
  description
    "Augmentation parameters apply only for networks with TE topology type.";
}

description
"Augment TE node tunnel termination point LLCs (Local Link Connectivities).";
uses te-node-tunnel-termination-point-llc-list;
}
</CODE ENDS>

8. Security Considerations

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

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

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

  This subtree specifies the TE topology type. Modifying the configurations can make TE topology type invalid. By such modifications, a malicious attacker may disable the TE capabilities on the related networks and cause traffic disrupted or misrouted.

- /nw:networks/tet:te
  This subtree specifies the TE node templates and TE link templates. Modifying the configurations in this subtree will change the related future TE configurations. By such modifications, a malicious attacker may change the TE capabilities scheduled at a future time, to cause traffic disrupted or misrouted.
o /nw:networks/nw:network
   This subtree specifies the topology-wide configurations, including
   the TE topology ID and topology-wide policies. Modifying the
   configurations in this subtree can add, remove, or modify TE
   topologies. By adding a TE topology, a malicious attacker may
   create an unauthorized traffic network. By removing or modifying a
   TE topology, a malicious attacker may cause traffic disabled or
   misrouted in the specified TE topology. Such traffic changes may
   also affect the traffic in the connected TE topologies.

o /nw:networks/nw:network/nw:node
   This subtree specifies the configurations for TE nodes. Modifying
   the configurations in this subtree can add, remove, or modify TE
   nodes. By adding a TE node, a malicious attacker may create an
   unauthorized traffic path. By removing or modifying a TE node, a
   malicious attacker may cause traffic disabled or misrouted in the
   specified TE node. Such traffic changes may also affect the
   traffic on the surrounding TE nodes and TE links in this TE
   topology and the connected TE topologies.

o /nw:networks/nw:network/nt:link/tet:te
   This subtree specifies the configurations for TE links. Modifying
   the configurations in this subtree can add, remove, or modify TE
   links. By adding a TE link, a malicious attacker may create an
   unauthorized traffic path. By removing or modifying a TE link, a
   malicious attacker may cause traffic disabled or misrouted on the
   specified TE link. Such traffic changes may also affect the
   traffic on the surrounding TE nodes and TE links in this TE
   topology and the connected TE topologies.

o /nw:networks/nw:network/nw:node/nt:termination-point
   This subtree specifies the configurations of TE link termination
   points. Modifying the configurations in this subtree can add,
   remove, or modify TE link termination points. By adding a TE link
   termination point, a malicious attacker may create an unauthorized
   traffic path. By removing or modifying a TE link termination
   point, a malicious attacker may cause traffic disabled or
   misrouted on the specified TE link termination point. Such traffic
   changes may also affect the traffic on the surrounding TE nodes
   and TE links in this TE topology and the connected TE topologies.

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

o /nw:networks/tet:te
  Unauthorized access to this subtree can disclose the TE node
  templates and TE link templates.

o /nw:networks/nw:network
  Unauthorized access to this subtree can disclose the topology-wide
  configurations, including the TE topology ID, the topology-wide
  policies, and the topology geolocation.

o /nw:networks/nw:network/nw:node
  Unauthorized access to this subtree can disclose the operational
  state information of TE nodes.

o /nw:networks/nw:network/nt:link/tet:te
  Unauthorized access to this subtree can disclose the operational
  state information of TE links.

o /nw:networks/nw:network/nw:node/nt:termination-point
  Unauthorized access to this subtree can disclose the operational
  state information of TE link termination points.

9. IANA Considerations

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

Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names
registry [RFC7950].

name:         ietf-te-topology
prefix:       tet
reference:    RFC XXXX
10. References

10.1. Normative References


10.2. Informative References


(GMPLS)", RFC 4203, DOI 10.17487/rfc4203, October 2005,

Hierarchy with Generalized Multi-Protocol Label Switching
(GMPLS) Traffic Engineering (TE)", RFC 4206,
DOI 10.17487/rfc4206, October 2005,

Ed., "RSVP-TE Extensions in Support of End-to-End
Generalized Multi-Protocol Label Switching (GMPLS)

Per-Domain Path Computation Method for Establishing Inter-
Domain Traffic Engineering (TE) Label Switched Paths
(LSPs)", RFC 5152, DOI 10.17487/rfc5152, February 2008,

[RFC5212] Shiomoto, K., Papadimitriou, D., Le Roux, JL., Vigoureux,
M., and D. Brungard, "Requirements for GMPLS-Based Multi-
Region and Multi-Layer Networks (MRN/MLN)", RFC 5212,
DOI 10.17487/rfc5212, July 2008,

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic
Engineering", RFC 5305, DOI 10.17487/rfc5305, October

[RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in
Support of Inter-Autonomous System (AS) MPLS and GMPLS
Traffic Engineering", RFC 5316, DOI 10.17487/rfc5316,

"Traffic Engineering Extensions to OSPF Version 3",
RFC 5329, DOI 10.17487/rfc5329, September 2008,

[RFC5392] Chen, M., Zhang, R., and X. Duan, "OSPF Extensions in
Support of Inter-Autonomous System (AS) MPLS and GMPLS
Traffic Engineering", RFC 5392, DOI 10.17487/rfc5392,

[RFC6001] Papadimitriou, D., Vigoureux, M., Shiomoto, K., Brungard,


11. Acknowledgments

The authors would like to thank Lou Berger, Sue Hares, Mazen Khaddam, Cyril Margaria and Zafar Ali for participating in design discussions and providing valuable insights.
Appendix A. Complete Model Tree Structure

module: ietf-te-topology
  augment /nw:networks/nw:network/nw:network-types:
    +++-rw te-topology!
augment /nw:networks:
    +++-rw te!
    +++-rw templates
      +++-rw node-template* [name] {template}?
        +++-rw name
          |  te-types:te-template-name
        +++-rw priority?  uint16
        +++-rw reference-change-policy?  enumeration
        +++-rw te-node-attributes
          +++-rw admin-status?  te-types:te-admin-status
          +++-rw domain-id?  uint32
          +++-rw is-abstract?  empty
          +++-rw name?  string
          +++-rw signaling-address*  inet:ip-address
          +++-rw underlay-topology {te-topology-hierarchy}?
            +++-rw network-ref?
              -> /nw:networks/network/network-id
      +++-rw link-template* [name] {template}?
        +++-rw name
          |  te-types:te-template-name
        +++-rw priority?  uint16
        +++-rw reference-change-policy?  enumeration
        +++-rw te-link-attributes
          +++-rw access-type?
            |  te-types:te-link-access-type
          +++-rw external-domain
            |  +++-rw network-ref?
            |    -> /nw:networks/network/network-id
            |    +++-rw remote-te-node-id?  te-types:te-node-id
            |    +++-rw remote-te-link-tp-id?  te-types:te-tp-id
          +++-rw is-abstract?  empty
          +++-rw name?  string
          +++-rw underlay {te-topology-hierarchy}?
            +++-rw enabled?  boolean
            +++-rw primary-path
              |  +++-rw network-ref?
-> /nw:networks/network/network-id
++rw path-element* [path-element-id]
   ++rw path-element-id          uint32
   ++rw (type)?
      ++-(numbered-node-hop)
         ++rw numbered-node-hop
            ++rw node-id     te-node-id
            ++rw hop-type?   te-hop-type
      ++-(numbered-link-hop)
         ++rw numbered-link-hop
            ++rw link-tp-id   te-tp-id
            ++rw hop-type?    te-hop-type
            ++rw direction?   te-link-direction
      ++-(unnumbered-link-hop)
         ++rw unnumbered-link-hop
            ++rw link-tp-id   te-tp-id
            ++rw node-id      te-node-id
            ++rw hop-type?     te-hop-type
            ++rw direction?    te-link-direction
      ++-(as-number)
         ++rw as-number-hop
            ++rw as-number   inet:as-number
            ++rw hop-type?   te-hop-type
      ++-(label)
         ++rw label-hop
            ++rw te-label
               ++rw (technology)?
                  ++-(generic)
                     ++rw generic?
                        rt-types:generalized-label
               ++rw direction?
                  te-label-direction
      ++rw backup-path* [index]
         ++rw index          uint32
         ++rw network-ref?
            -> /nw:networks/network/network-id
         ++rw path-element* [path-element-id]
            ++rw path-element-id  uint32
++rw admin-status?
  |  te-types:te-admin-status
++rw link-index?  uint64
++rw administrative-group?
  |  te-types:admin-groups
++rw interface-switching-capability*
  [switching-capability encoding]
    ++rw switching-capability  identityref
    ++rw encoding  identityref
++rw max-lsp-bandwidth* [priority]
    ++rw priority  uint8
    ++rw te-bandwidth
    ++rw (technology)?
      ++: (generic)
        ++rw generic?  te-bandwidth
++rw label-restrictions
  ++rw label-restriction* [index]
    ++rw restriction?  enumeration
    ++rw index  uint32
    ++rw label-start
      ++rw te-label
        ++rw (technology)?
          ++: (generic)
            ++rw generic?
              rt-types:generalized-label
        ++rw direction?  te-label-direction
    ++rw label-end
      ++rw te-label
        ++rw (technology)?
          ++: (generic)
            ++rw generic?
              rt-types:generalized-label
        ++rw direction?  te-label-direction
    ++rw label-step
      ++rw (technology)?
        ++: (generic)
          ++rw generic?  int32
            ++rw range-bitmap?  yang:hex-string
++rw link-protection-type?  identityref
++rw max-link-bandwidth
  |  ++rw te-bandwidth
---rw max-resv-link-bandwidth
  +--rw te-bandwidth
---rw (technology)?
  +--:(generic)
    +--rw generic?  te-bandwidth
---rw unreserved-bandwidth* [priority]
  +--rw priority  uint8
  +--rw te-bandwidth
---rw (technology)?
  +--:(generic)
    +--rw generic?  te-bandwidth
---rw te-default-metric?  uint32
---rw te-delay-metric?  uint32
---rw te-igp-metric?  uint32
---rw te-srlgs
  +--rw value*  te-types:srlg
  +--rw te-nsrlgs {nsrlg}?
    +--rw id*  uint32

 augment /nw:networks/nw:network:
  +--rw te-topology-identifier
    +--rw provider-id?  te-global-id
    +--rw client-id?  te-global-id
    +--rw topology-id?  te-topology-id
  +--rw te!
    +--rw name?  string
    +--rw preference?  uint8
    +--rw optimization-criterion?  identityref
    +--rw nsrlg* [id] {nsrlg}?
      +--rw id  uint32
      +--rw disjointness?  te-types:te-path-disjointness
    +--ro geolocation
      +--ro altitude?  int64
      +--ro latitude?  geographic-coordinate-degree
      +--ro longitude?  geographic-coordinate-degree

 augment /nw:networks/nw:network/nw:node:
  +--rw te-node-id?  te-types:te-node-id
  +--rw te!
    +--rw te-node-template*
Internet-Draft            YANG - TE Topology                  June 2019

-> ../../../te/templates/node-template/name
   {template}?
---rw te-node-attributes
  +--rw admin-status? te-types:te-admin-status
---rw connectivity-matrices
  +--rw number-of-entries? uint16
  +--rw label-restrictions
    +--rw label-restriction* [index]
      +--rw restriction? enumeration
      +--rw index uint32
    +--rw label-start
      +--rw te-label
        +--rw (technology)?
          +--:(generic)
            +--rw generic? rt-types:generalized-label
        +--rw direction? te-label-direction
    +--rw label-end
      +--rw te-label
        +--rw (technology)?
          +--:(generic)
            +--rw generic? rt-types:generalized-label
        +--rw direction? te-label-direction
    +--rw label-step
      +--rw (technology)?
        +--:(generic)
          +--rw generic? int32
      +--rw range-bitmap? yang:hex-string
---rw is-allowed? boolean
---rw underlay {te-topology-hierarchy}?
  +--rw enabled? boolean
  +--rw primary-path
    +--rw network-ref?
      -> /nw:networks/network/network-id
    +--rw path-element* [path-element-id]
      +--rw path-element-id uint32
      +--rw (type)?
        +--:(numbered-node-hop)
          +--rw numbered-node-hop
            +--rw node-id te-node-id

+--rw hop-type?  te-hop-type
  +--:(numbered-link-hop)
  |   +--rw numbered-link-hop
  |      +--rw link-tp-id   te-tp-id
  |      +--rw hop-type?    te-hop-type
  |      +--rw direction?   te-link-direction
  +--:(unnumbered-link-hop)
  |   +--rw unnumbered-link-hop
  |      +--rw link-tp-id   te-tp-id
  |      +--rw node-id     te-node-id
  |      +--rw hop-type?    te-hop-type
  |      +--rw direction?   te-link-direction
  +--:(as-number)
  |   +--rw as-number-hop
  |      +--rw as-number    inet:as-number
  |      +--rw hop-type?    te-hop-type
  +--:(label)
  |   +--rw label-hop
  |      +--rw te-label
  |      +--rw (technology)?
  |         +--:(generic)
  |            +--rw generic?
  |               rt-types:generalized-label
  +--rw backup-path* [index]
  |   +--rw index       uint32
  |       -> /nw:networks/network/network-id
  +--rw path-element* [path-element-id]
  |   +--rw path-element-id       uint32
  |   +--rw (type)?
  |      +--:(numbered-node-hop)
  |         +--rw numbered-node-hop
  |            +--rw node-id     te-node-id
  |            +--rw hop-type?   te-hop-type
  +--:(numbered-link-hop)
  |   +--rw numbered-link-hop
  |      +--rw link-tp-id   te-tp-id
  |      +--rw hop-type?    te-hop-type
++--rw direction? te-link-direction
++--(unnumbered-link-hop)
  ++--rw unnumbered-link-hop
    ++--rw link-tp-id te-tp-id
    ++--rw node-id te-node-id
    ++--rw hop-type? te-hop-type
    ++--rw direction? te-link-direction
++--(as-number)
  ++--rw as-number-hop
    ++--rw as-number inet:as-number
    ++--rw hop-type? te-hop-type
++--(label)
  ++--rw label-hop
    ++--rw te-label
      ++--rw (technology)?
        ++--(generic)
          ++--rw generic?
            rt-types:generalized-label
    ++--rw direction? te-label-direction
++--rw protection-type? identityref
++--rw tunnel-termination-points
  ++--rw source? binary
  ++--rw destination? binary
++--rw tunnels
  ++--rw sharing? boolean
  ++--rw tunnel* [tunnel-name]
    ++--rw tunnel-name string
    ++--rw sharing? boolean
++--rw path-constraints
  ++--rw te-bandwidth
    ++--rw (technology)?
      ++--(generic)
        ++--rw generic? te-bandwidth
    ++--rw link-protection? identityref
    ++--rw setup-priority? uint8
    ++--rw hold-priority? uint8
    ++--rw signaling-type? identityref
  ++--rw path-metric-bounds
    ++--rw path-metric-bound* [metric-type]
---rw metric-type identityref
---rw upper-bound? uint64
+++rw path-affinities-values
    +++rw path-affinities-value* [usage]
    ++-rw usage identityref
    +++rw value? admin-groups
+++rw path-affinity-names
    +++rw path-affinity-name* [usage]
    ++-rw usage identityref
    +++rw affinity-name* [name]
    +++rw name string
+++rw path-srlgs-lists
    +++rw path-srlgs-list* [usage]
    ++-rw usage identityref
    +++rw values* srlg
+++rw path-srlgs-names
    +++rw path-srlgs-name* [usage]
    ++-rw usage identityref
    +++rw names* string
+++rw disjointness? te-path-disjointness
+++rw optimizations
+++rw (algorithm)?
    +++: (metric) {path-optimization-metric}? 
    +++rw optimization-metric* [metric-type]
    ++-rw metric-type
        identityref
    ++-rw weight? uint8
    +++rw explicit-route-exclude-objects
        +++rw route-object-exclude-object* [index]
        ++-rw index uint32
    +++rw (type)?
        +++: (numbered-node-hop) 
        +++rw numbered-node-hop
            +++rw node-id te-node-id
            +++rw hop-type? te-hop-type
        +++: (numbered-link-hop) 
        +++rw numbered-link-hop
            +++rw link-tp-id te-tp-id
++--rw hop-type?
    |    te-hop-type
++--rw direction?
    |    te-link-direction
+++:(unnumbered-link-hop)
+++--rw unnumbered-link-hop
    +++--rw link-tp-id    te-tp-id
    +++--rw node-id
        |    te-node-id
    +++--rw hop-type?
        |    te-hop-type
    +++--rw direction?
        |    te-link-direction
+++:(as-number)
+++--rw as-number-hop
    +++--rw as-number
        |    inet:as-number
    +++--rw hop-type?
        |    te-hop-type
+++:(label)
+++--rw label-hop
    +++--rw te-label
        +++--rw (technology)?
            +++--:(generic)
                +++--rw generic?
                    rt-
    types:generalized-label
    +++--rw direction?
        te-label-direction
+++:(srlg)
+++--rw srlg
    +++--rw srlg?    uint32
    +++--rw explicit-route-include-objects
        +++--rw route-object-include-object*
            [index]
    +++--rw index
        |    uint32
    +++--rw (type)?
        +++--:(numbered-node-hop)
            +++--rw numbered-node-hop
                +++--rw node-id    te-node-id
---rw hop-type?  te-hop-type
+-:(numbered-link-hop)
  ---rw numbered-link-hop
  +---rw link-tp-id  te-tp-id
  +---rw hop-type?
      |  te-hop-type
  +---rw direction?
      te-link-direction
+-:(unnumbered-link-hop)
  ---rw unnumbered-link-hop
  +---rw link-tp-id  te-tp-id
  +---rw node-id
      |  te-node-id
  +---rw hop-type?
      |  te-hop-type
  +---rw direction?
      te-link-direction
+-:(as-number)
  ---rw as-number-hop
  +---rw as-number
      |  inet:as-number
  +---rw hop-type?
      te-hop-type
+-:(label)
  ---rw label-hop
  +---rw te-label
      +---rw (technology)?
          +---:(generic)
              +---rw generic?
                  rt-
  types:generalized-label
  +---rw direction?
      te-label-direction
  +---rw tiebreakers
      +---rw tiebreaker* [tiebreaker-type]
  +---rw tiebreaker-type  identityref
+-:(objective-function)
  {path-optimization-objective-function}?
  +---rw objective-function
  +---rw objective-function-type?  identityref
+-ro path-properties
Internet-Draft            YANG - TE Topology                  June 2019

---ro as-number           inet:as-number
  +--ro hop-type?         te-hop-type
  +--:(label)
    +--ro label-hop
    +--ro te-label
      +--ro (technology)?
      +--:(generic)
        +--ro generic?
          rt-types:generalized-

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

---rw connectivity-matrix* [id]
  +--rw id                uint32
  +--rw from
    +--rw tp-ref?          leafref
    +--rw label-restrictions
      +--rw label-restriction* [index]
        +--rw restriction?    enumeration
        +--rw index           uint32
      +--rw label-start
        +--rw te-label
          +--rw (technology)?
          +--:(generic)
            +--rw generic?
              rt-types:generalized-

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

---rw label-end
  +--rw te-label
    +--rw (technology)?
    +--:(generic)
      +--rw generic?
        rt-types:generalized-

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

---rw label-step
  +--rw (technology)?
  +--:(generic)
++-rw generic?  int32
++-rw range-bitmap?  yang:hex-string
+-rw to
  +-rw tp-ref?  leafref
  +-rw label-restrictions
    +-rw label-restriction*  [index]
      +-rw restriction?  enumeration
      +-rw index  uint32
    +-rw label-start
      +-rw te-label
        +-rw (technology)?
        |  +-rw generic?
        |    |  rt-types:generalized-label
        |  +-rw direction?
        |    te-label-direction
    +-rw label-end
      +-rw te-label
        +-rw (technology)?
        |  +-rw generic?
        |    |  rt-types:generalized-label
        |  +-rw direction?
        |    te-label-direction
      +-rw (technology)?
      |  +-rw generic?  int32
        |    |  rt-types:generalized-label
      +-rw range-bitmap?  yang:hex-string
    +-rw is-allowed?  boolean
    +-rw underlay {te-topology-hierarchy}?
      +-rw enabled?  boolean
      +-rw primary-path
        +-rw network-ref?
          |  -> /nw:networks/network/network-id
        +-rw path-element*  [path-element-id]
          +-rw path-element-id  uint32
          +-rw (type)?
            |  +-rw (numbered-node-hop)
```yang
++-rw numbered-node-hop
    +-rw node-id     te-node-id
    +-rw hop-type?   te-hop-type
+-:(numbered-link-hop)
  +-rw numbered-link-hop
    +-rw link-tp-id    te-tp-id
    +-rw hop-type?   te-hop-type
    +-rw direction?
      te-link-direction
  +-:(unnumbered-link-hop)
    +-rw unnumbered-link-hop
      +-rw link-tp-id    te-tp-id
      +-rw node-id       te-node-id
      +-rw hop-type?   te-hop-type
      +-rw direction?
        te-link-direction
  +-:(as-number)
    +-rw as-number-hop
      +-rw as-number    inet:as-number
      +-rw hop-type?   te-hop-type
  +-:(label)
    +-rw label-hop
      +-rw te-label
        +-rw (technology)?
          +-:(generic)
            +-rw generic?
              rt-types:generalized-label
                +-rw direction?
                  te-label-direction
          +-rw backup-path* [index]
            +-rw index        uint32
              -> /nw:networks/network/network-id
            +-rw path-element* [path-element-id]
              +-rw path-element-id        uint32
              +-rw (type)?
                +-:(numbered-node-hop)
                  +-rw numbered-node-hop
                    +-rw node-id     te-node-id
                    +-rw hop-type?   te-hop-type
```
| +--:(numbered-link-hop) |
| +--rw numbered-link-hop |
| +--rw link-tp-id         te-tp-id |
| +--rw hop-type?          te-hop-type |
| +--rw direction?         te-link-direction |

| +--:(unnumbered-link-hop) |
| +--rw unnumbered-link-hop |
| +--rw link-tp-id         te-tp-id |
| +--rw node-id            te-node-id |
| +--rw hop-type?          te-hop-type |
| +--rw direction?         te-link-direction |

| +--:(as-number) |
| +--rw as-number-hop |
| +--rw as-number       inet:as-number |
| +--rw hop-type?       te-hop-type |

| +--:(label) |
| +--rw label-hop |
| +--rw te-label |
| +--rw (technology)? |
| | +--:(generic) |
| | | +--rw generic? |

| types:generalized-label |
| +--rw direction?        te-label-direction |
| +--rw protection-type?  identityref |

| +--rw tunnel-termination-points |
| +--rw source?             binary |
| +--rw destination?        binary |
| +--rw tunnels |
| +--rw sharing?            boolean |
| +--rw tunnel*            [tunnel-name] |
| +--rw tunnel-name         string |

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

direction
    ++-rw tiebreakers
        ++-rw tiebreaker* [tiebreaker-type]
            ++-rw tiebreaker-type identityref
    +--:(objective-function)
        (path-optimization-objective-function)?
            ++-rw objective-function
                ++-rw objective-function-type?
                    identityref
    ++-ro path-properties
        ++-ro path-metric* [metric-type]
            ++-ro metric-type identityref
            ++-ro accumulative-value? uint64
        ++-ro path-affinities-values
            ++-ro path-affinities-value* [usage]
                ++-ro usage identityref
                ++-ro value? admin-groups
        ++-ro path-affinity-names
            ++-ro path-affinity-name* [usage]
                ++-ro usage identityref
                ++-ro affinity-name* [name]
                    ++-ro name string
        ++-ro path-srlgs-lists
            ++-ro path-srlgs-list* [usage]
                ++-ro usage identityref
                ++-ro values* srlg
        ++-ro path-srlgs-names
            ++-ro path-srlgs-name* [usage]
                ++-ro usage identityref
                ++-ro names* string
        ++-ro path-route-objects
            ++-ro path-route-object* [index]
---ro index            uint32
---ro (type)?
  ---:(numbered-node-hop)
    ---ro numbered-node-hop
      ---ro node-id     te-node-id
      ---ro hop-type?   te-hop-type
  ---:(numbered-link-hop)
    ---ro numbered-link-hop
      ---ro link-tp-id  te-tp-id
      ---ro hop-type?   te-hop-type
      ---ro direction?  te-link-direction
  ---:(unnumbered-link-hop)
    ---ro unnumbered-link-hop
      ---ro link-tp-id  te-tp-id
      ---ro node-id     te-node-id
      ---ro hop-type?   te-hop-type
      ---ro direction?  te-link-direction
  ---:(as-number)
    ---ro as-number-hop
      ---ro as-number   inet:as-number
      ---ro hop-type?   te-hop-type
  ---:(label)
    ---ro label-hop
      ---ro te-label
      ---ro (technology)?
        ---:(generic)
          ---ro generic?
            rt-types:generalized-label
      ---ro direction?  te-label-direction
    ---rw domain-id?    uint32
    ---rw is-abstract?  empty
    ---rw name?         string
    ---rw signaling-address* inet:ip-address
    ---rw underlay-topology {te-topology-hierarchy}? 
    ---ro oper-status?   te-types:te-oper-status
    ---ro geolocation
| +--ro altitude?  int64
| +--ro latitude?  geographic-coordinate-degree
| +--ro longitude?  geographic-coordinate-degree
+--ro is-multi-access-dr?  empty
+--ro information-source?  te-info-source
+--ro information-source-instance?  string
+--ro information-source-state
  +--ro credibility-preference?  uint16
  +--ro logical-network-element?  string
  +--ro network-instance?  string
  +--ro topology
    +--ro node-ref?  leafref
+--ro information-source-entry*
  [information-source information-source-instance]
    +--ro information-source  te-info-source
    +--ro information-source-instance  string
+--ro visibility
  +--ro credibility-preference?  uint16
  +--ro logical-network-element?  string
  +--ro network-instance?  string
  +--ro topology
    +--ro node-ref?  leafref
+--ro connectivity-matrices
  +--ro number-of-entries?  uint16
  +--ro label-restrictions
    +--ro label-restriction* [index]
      +--ro restriction?  enumeration
      +--ro index  uint32
    +--ro label-start
      +--ro te-label
        +--ro (technology)?
          +--:(generic)
            +--ro generic?
              rt-types:generalized-label
            +--ro direction?  te-label-direction
    +--ro label-end
      +--ro te-label
      +--ro (technology)?
++-:(generic)
  ++-ro generic?
    rt-types:generalized-label
      ++-ro direction?  te-label-direction
  ++-ro label-step
    ++-ro (technology)?
      ++-:(generic)
        ++-ro generic?  int32
      ++-ro range-bitmap?  yang:hex-string
  ++-ro is-allowed?  boolean
++-ro underlay {te-topology-hierarchy}?
  ++-ro enabled?  boolean
++-ro primary-path
  ++-ro network-ref?
    -> /nw:networks/network/network-id
++-ro path-element* [path-element-id]
  ++-ro path-element-id  uint32
++-ro (type)?
  ++-:(numbered-node-hop)
    ++-ro numbered-node-hop
      ++-ro node-id  te-node-id
      ++-ro hop-type?  te-hop-type
  ++-:(numbered-link-hop)
    ++-ro numbered-link-hop
      ++-ro link-tp-id  te-tp-id
      ++-ro hop-type?  te-hop-type
      ++-ro direction?  te-link-direction
  ++-:(unnumbered-link-hop)
    ++-ro unnumbered-link-hop
      ++-ro link-tp-id  te-tp-id
      ++-ro node-id  te-node-id
      ++-ro hop-type?  te-hop-type
      ++-ro direction?  te-link-direction
  ++-:(as-number)
    ++-ro as-number-hop
      ++-ro as-number  inet:as-number
      ++-ro hop-type?  te-hop-type
  ++-:(label)
    ++-ro label-hop
    ++-ro te-label
      ++-ro (technology)?
te-label-direction

---ro protection-type?
    identityref

---ro tunnel-termination-points
    +--ro source?
        binary
    +--ro destination?
        binary

---ro tunnels
    +--ro sharing?
        boolean
    +--ro tunnel* [tunnel-name]
        +--ro tunnel-name
            string
        +--ro sharing?
            boolean

---ro path-constraints
    +--ro te-bandwidth
        +--ro (technology)?
            +--:(generic)
                +--ro generic?
                    te-bandwidth
        +--ro link-protection?
            identityref
        +--ro setup-priority?
            uint8
        +--ro hold-priority?
            uint8
        +--ro signaling-type?
            identityref

---ro path-metric-maps
    +--ro path-metric-bound* [metric-type]
        +--ro metric-type
            identityref
        +--ro upper-bound?
            uint64

---ro path-affinities-values
    +--ro path-affinities-value* [usage]
        +--ro usage
            identityref
        +--ro value?
            admin-groups

---ro path-affinity-names
    +--ro path-affinity-name* [usage]
        +--ro usage
            identityref
        +--ro affinity-name* [name]
            +--ro name
                string

---ro path-srlgs-lists
    +--ro path-srlgs-list* [usage]
        +--ro usage
            identityref
        +--ro values* [srlg]

---ro path-srlgs-names
    +--ro path-srlgs-name* [usage]
        +--ro usage
            identityref
        +--ro names* [name]

---ro disjointness?
    te-path-disjointness
---ro optimizations
  ---ro (algorithm)?
    ---:(metric) {path-optimization-metric}?  
      ---ro optimization-metric* [metric-type]
      |    ---ro metric-type
      |         identityref
      |         uint8
      ---ro explicit-route-exclude-objects
        ---ro route-object-exclude-object* [index]
          ---ro index
            |         uint32
      ---ro (type)?
        ---:(numbered-node-hop)
          ---ro numbered-node-hop
            |        ---ro node-id     te-node-id
            |        ---ro hop-type?  te-hop-type
        ---:(numbered-link-hop)
          ---ro numbered-link-hop
            |      ---ro link-tp-id    te-tp-id
            |      ---ro hop-type?
            |        te-hop-type
            |      ---ro direction?
            |        te-link-direction
        ---:(unnumbered-link-hop)
          ---ro unnumbered-link-hop
            |      ---ro link-tp-id    te-tp-id
            |      ---ro node-id
            |        te-node-id
            |      ---ro hop-type?
            |        te-hop-type
            |      ---ro direction?
            |        te-link-direction
        ---:(as-number)
          ---ro as-number-hop
            |          ---ro as-number
            |           inet:as-number
            |          ---ro hop-type?
            |           te-hop-type
        ---:(label)
++-ro path-route-objects
  +++-ro path-route-object* [index]
    +++-ro index uint32
    +++-ro (type)?
      +++-:(numbered-node-hop)
        +++-ro numbered-node-hop
          +++-ro node-id te-node-id
          +++-ro hop-type? te-hop-type
      +++-:(numbered-link-hop)
        +++-ro numbered-link-hop
          +++-ro link-tp-id te-tp-id
          +++-ro hop-type? te-hop-type
          +++-ro direction? te-link-direction
      +++-:(unnumbered-link-hop)
        +++-ro unnumbered-link-hop
          +++-ro link-tp-id te-tp-id
          +++-ro node-id te-node-id
          +++-ro hop-type? te-hop-type
          +++-ro direction? te-link-direction
      +++-:(as-number)
        +++-ro as-number-hop
          +++-ro as-number inet:as-number
          +++-ro hop-type? te-hop-type
      +++-:(label)
        +++-ro label-hop
          +++-ro te-label
            +++-ro (technology)?
              +++-:(generic)
                +++-ro generic?
                  rt-types:generalized-
                  label
                  +++-ro direction?
                    te-label-direction
          +++-ro connectivity-matrix* [id]
            +++-ro id uint32
            +++-ro from
              +++-ro tp-ref? leafref
              +++-ro label-restrictions
                +++-ro label-restriction* [index]
                  +++-ro restriction? enumeration
                  +++-ro index uint32
+-ro label-start
  +-ro te-label
    +-ro (technology)?
      +-ro generic?
        rt-types:generalized-
label
  +-ro direction?
    te-label-direction
+-ro label-end
  +-ro te-label
    +-ro (technology)?
      +-ro generic?
        rt-types:generalized-
label
  +-ro direction?
    te-label-direction
  +-ro label-step
    +-ro (technology)?
      +-ro generic?  int32
        +-ro generic?  int32
  +-ro range-bitmap?  yang:hex-string
+-ro to
  +-ro tp-ref?  leafref
    +-ro label-restrictions
      +-ro label-restriction* [index]
        +-ro restriction?  enumeration
        +-ro index  uint32
    +-ro label-start
      +-ro te-label
        +-ro (technology)?
          +-ro generic?
          rt-types:generalized-
label
  +-ro direction?
    te-label-direction
+-ro label-end
  +-ro te-label
    +-ro (technology)?
label

- ro direction?
  te-label-direction
- ro label-step
- ro (technology)?
  - ro generic? int32
  - ro range-bitmap? yang:hex-string
- ro is-allowed? boolean
- ro underlay {te-topology-hierarchy}? boolean
- ro primary-path
  - ro network-ref?
    - > /nw:networks/network/network-id
  - ro path-element* [path-element-id]
    - ro path-element-id uint32
    - ro (type)?
      - : (numbered-node-hop)
        - ro numbered-node-hop
          - ro node-id te-node-id
          - ro hop-type? te-hop-type
      - : (numbered-link-hop)
        - ro numbered-link-hop
          - ro link-tp-id te-tp-id
          - ro hop-type? te-hop-type
          - ro direction?
            te-link-direction
      - : (unnumbered-link-hop)
        - ro unnumbered-link-hop
          - ro link-tp-id te-tp-id
          - ro node-id te-node-id
          - ro hop-type? te-hop-type
          - ro direction?
            te-link-direction
      - : (as-number)
        - ro as-number-hop
          - ro as-number inet:as-number
          - ro hop-type? te-hop-type
```
| | | | | +--:(label)
| | | | +--ro label-hop
| | | +--ro te-label
| | +--ro (technology)?
| | | +--:(generic)
| | | | +--ro generic?
| | | | | rt-
| | | | types:generalized-label
| | | +--ro direction?
| | | | te-label-direction
| | | +--ro backup-path* [index]
| | | | +--ro index uint32
| | | | +--ro network-ref?
| | | | | -> /nw:networks/network/network-id
| | | +--ro path-element* [path-element-id]
| | | | +--ro path-element-id uint32
| | | +--ro (type)?
| | | | +--:(numbered-node-hop)
| | | | | +--ro number-node-hop
| | | | | | +--ro node-id te-node-id
| | | | | | +--ro hop-type? te-hop-type
| | | | +--:(numbered-link-hop)
| | | | | +--ro number-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
```

```yang
module te-topology {
  include ietf-yang-types;

  leaf srlg {
    leaf path-srlgs-names {
      leaf path-srlgs-name {
        leaf usage {
          leaf names {
            leaf disjointness {
              leaf (te-path-disjointness)
            }
            leaf optimizations {
              leaf (algorithm) {
                leaf (metric) {
                  leaf (optimization-metric) {
                    leaf metric-type {
                      leaf (metric-type) {
                        leaf weight {
                          leaf (uint8)
                        }
                        leaf explicit-route-exclude-objects {
                          leaf route-object-exclude-object {
                            leaf index {
                              leaf (uint32)
                            }
                          }
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```

```yang
---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
         ---:(technology)?
            ---ro generic?
               rt-types:generalized-label
   ---ro direction?
      te-label-direction
---:(srlg)
   ---ro srlg?
   ---ro srlg? uint32
---ro explicit-route_include_objects
   ---ro route_object_include_object*
      [index]
      ---ro index
         uint32
   ---ro (type)?
      ---:(numbered-node-hop)
         ---ro numbered-node-hop
            ---ro node-id
               te-node-id
            ---ro hop-type?
               te-hop-type
      ---:(numbered-link-hop)
         ---ro numbered-link-hop
            ---ro link-tp-id
               te-tp-id
            ---ro hop-type?
```

Liu, et al  Expires December 19, 2019  [Page 135]
te-hop-type
  +--ro direction?
  te-link-direction
  +--:(unnumbered-link-hop)
    +--ro unnumbered-link-hop
      +--ro link-tp-id
      |  te-tp-id
      +--ro node-id
      |  te-node-id
      +--ro hop-type?
      |  te-hop-type
      +--ro direction?
      |  te-link-direction
  +--:(as-number)
    +--ro as-number-hop
      +--ro as-number
      |  inet:as-number
      +--ro hop-type?
      |  te-hop-type
  +--:(label)
    +--ro label-hop
      +--ro te-label
      +--ro (technology)?
      |  +--:(generic)
      |       +--ro generic?
      |         rt-types:generalized-label
      |  +--ro direction?
      |   te-label-

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

<table>
<thead>
<tr>
<th>types:generalized-label</th>
</tr>
</thead>
<tbody>
<tr>
<td>++-ro direction?</td>
</tr>
<tr>
<td>te-label-direction</td>
</tr>
<tr>
<td>+--ro domain-id?          uint32</td>
</tr>
<tr>
<td>+--ro is-abstract?        empty</td>
</tr>
<tr>
<td>+--ro name?               string</td>
</tr>
<tr>
<td>+--ro signaling-address*  inet:ip-address</td>
</tr>
<tr>
<td>+--ro underlay-topology   (te-topology-hierarchy)?</td>
</tr>
</tbody>
</table>
++-ro statistics
  +--ro discontinuity-time? yang:date-and-time
++-ro node
  +--ro disables?           yang:counter32
  +--ro enables?            yang:counter32
  +--ro maintenance-sets?   yang:counter32
  +--ro maintenance-clears? yang:counter32
  +--ro modifies?           yang:counter32
  +--ro connectivity-matrix-entry
    +--ro creates?         yang:counter32
    +--ro deletes?         yang:counter32
    +--ro disables?        yang:counter32
    +--ro enables?         yang:counter32
    +--ro modifies?        yang:counter32
++-rw tunnel-termination-point* [tunnel-tp-id]
  +--rw tunnel-tp-id        binary
  +--rw admin-status?
    |    te-types:te-admin-status
    |    +--rw name?               string
    |    +--rw switching-capability? identityref
    |    +--rw encoding?           identityref
    |    +--rw inter-layer-lock-id* uint32
++-rw protection-type?                      identityref
  ++-rw client-layer-adaptation
    ++-rw switching-capability* [switching-capability encoding]
      ++-rw switching-capability identityref
      ++-rw encoding identityref
      ++-rw te-bandwidth
        ++-rw (technology)?
          +--:(generic)
            ++-rw generic? te-bandwidth
  ++-rw local-link-connectivities
    ++-rw number-of-entries? uint16
    ++-rw label-restrictions
      ++-rw label-restriction* [index]
        ++-rw restriction? enumeration
        ++-rw index uint32
        ++-rw label-start
          ++-rw te-label
            ++-rw (technology)?
              +--:(generic)
                ++-rw generic? rt-types:generalized-label
            ++-rw direction? te-label-direction
        ++-rw label-end
        ++-rw te-label
          ++-rw (technology)?
            +--:(generic)
              ++-rw generic? rt-types:generalized-label
          ++-rw direction? te-label-direction
        ++-rw label-step
          ++-rw (technology)?
            +--:(generic)
              ++-rw generic? int32
        ++-rw range-bitmap? yang:hex-string
    ++-rw is-allowed? boolean
    ++-rw underlay {te-topology-hierarchy}? |
      ++-rw enabled? boolean
      ++-rw primary-path |
      ++-rw network-ref? |
        -> /nw:networks/network/network-id
---rw path-element* [path-element-id]
  ---rw path-element-id uint32
  ---rw (type)?
    +++:(numbered-node-hop)
    |  ---rw numbered-node-hop
    |     ---rw node-id te-node-id
    |     ---rw hop-type? te-hop-type
    +++:(numbered-link-hop)
    |  ---rw numbered-link-hop
    |     ---rw link-tp-id te-tp-id
    |     ---rw hop-type? te-hop-type
    |     ---rw direction? te-link-direction
    +++:(unnumbered-link-hop)
    |  ---rw unnumbered-link-hop
    |     ---rw link-tp-id te-tp-id
    |     ---rw node-id te-node-id
    |     ---rw hop-type? te-hop-type
    |     ---rw direction? te-link-direction
    +++:(as-number)
    |  ---rw as-number-hop
    |     ---rw as-number inet:as-number
    |     ---rw hop-type? te-hop-type
    +++:(label)
    |  ---rw label-hop
    |     ---rw te-label
    |        ---rw (technology)?
    |        +++:(generic)
    |        |  ---rw generic?
    |        |     rt-types:generalized-label
    |     ---rw direction? te-label-direction
  ---rw backup-path* [index]
    ---rw index uint32
    ---rw network-ref?
    |      -> /nw:networks/network/network-id
    ---rw path-element* [path-element-id]
    |  ---rw path-element-id uint32
    |  ---rw (type)?
    |     +++:(numbered-node-hop)
    |     |  ---rw numbered-node-hop
++-rw node-id  te-node-id
++-rw hop-type?  te-hop-type
+-:(numbered-link-hop)
  ++-rw numbered-link-hop
    ++-rw link-tp-id  te-tp-id
    ++-rw hop-type?  te-hop-type
    ++-rw direction?  te-link-direction
  ++-:(unnumbered-link-hop)
    ++-rw unnumbered-link-hop
      ++-rw link-tp-id  te-tp-id
      ++-rw node-id  te-node-id
      ++-rw hop-type?  te-hop-type
      ++-rw direction?  te-link-direction
    ++-:(as-number)
      ++-rw as-number-hop
        ++-rw as-number  inet:as-number
        ++-rw hop-type?  te-hop-type
    ++-:(label)
      ++-rw label-hop
        ++-rw te-label
          ++-rw (technology)?
            ++-:(generic)
              ++-rw generic?
                rt-types:generalized-label
        ++-rw direction?
          te-label-direction
      ++-rw protection-type?
        identityref
  ++-rw tunnel-termination-points
    ++-rw source?  binary
    ++-rw destination?  binary
  ++-rw tunnels
    ++-rw sharing?  boolean
    ++-rw tunnel*  [tunnel-name]
      ++-rw tunnel-name  string
      ++-rw sharing?  boolean
  ++-rw path-constraints
    ++-rw te-bandwidth
      ++-rw (technology)?
        ++-:(generic)
          ++-rw generic?  te-bandwidth
++-rw link-protection? identityref
++-rw setup-priority? uint8
++-rw hold-priority? uint8
++-rw signaling-type? identityref
++-rw path-metric-bounds
  |  +--rw path-metric-bound* [metric-type]
  |     +--rw metric-type identityref
  |     +--rw upper-bound? uint64
++-rw path-affinities-values
  |  +--rw path-affinities-value* [usage]
  |     +--rw usage identityref
  |     +--rw value? admin-groups
++-rw path-affinity-names
  |  +--rw path-affinity-name* [usage]
  |     +--rw usage identityref
  |     +--rw affinity-name* [name]
  |        +--rw name string
++-rw path-srlgs-lists
  |  +--rw path-srlgs-list* [usage]
  |     +--rw usage identityref
  |     +--rw values* srlg
++-rw path-srlgs-names
  |  +--rw path-srlgs-name* [usage]
  |     +--rw usage identityref
  |     +--rw names* string
++-rw disjointness? te-path-disjointness
++-rw optimizations
  +--rw (algorithm)?
    |  +--:(metric) {path-optimization-metric}?
    |     +--rw optimization-metric* [metric-type]
    |        +--rw metric-type identityref
    |        +--rw weight? uint8
    |        +--rw explicit-route-exclude-objects
    |             +--rw route-object-exclude-object* [index]
    |                 +--rw index uint32
    |                 +--rw (type)?
    |                    +--:(numbered-node-hop)
---rw numbered-node-hop
   ---rw node-id  te-node-id
   ---rw hop-type? te-hop-type
   ++-:(numbered-link-hop)
   ++-rw numbered-link-hop
      ---rw link-tp-id  te-tp-id
      ---rw hop-type?  te-hop-type
      ---rw direction?  te-link-direction
      ++-:(unnumbered-link-hop)
      ++-rw unnumbered-link-hop
         ---rw link-tp-id  te-tp-id
         ++-rw node-id  te-node-id
         ---rw hop-type?  te-hop-type
         ---rw direction?  te-link-direction
   ++-:(as-number)
   ++-rw as-number-hop
      ---rw as-number
         |  inet:as-number
         ++-rw hop-type?  te-hop-type
   ++-:(label)
   ++-rw label-hop
      ---rw te-label
         ++-rw (technology)?
            ++-:(generic)
               ++-rw generic?
                  rt-
         ++-rw direction?  te-label-direction
   ++-:(srlg)
   ++-rw srlg
      ---rw srlg?  uint32
      ++-rw explicit-route-include-objects
         ++-rw route-object-include-object*
            [index]

types:generalized-label
++--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
            ++--rw inet:as-number
         ++--rw hop-type?    te-hop-type
   ++--:(label)
      ++--rw label-hop
         ++--rw te-label
            ++--rw (technology)?
               ++--:(generic)
                  ++--rw generic?
                      rt-types:generalized-label
                         ++--rw direction?
                            te-label-direction
   ++--rw tiebreakers
      ++--rw tiebreaker* [tiebreaker-type]
---rw tiebreaker-type  identityref

---:(objective-function)

{path-optimization-objective-function)?

---rw objective-function

---rw objective-function-type?  identityref

---ro path-properties

---ro path-metric*  [metric-type]

---ro metric-type  identityref

---ro accumulative-value?  uint64

---ro path-affinities-values

---ro path-affinities-value*  [usage]

---ro usage  identityref

---ro value?  admin-groups

---ro path-affinity-names

---ro path-affinity-name*  [usage]

---ro usage  identityref

---ro affinity-name*  [name]

---ro name  string

---ro path-srlgs-lists

---ro path-srlgs-list*  [usage]

---ro usage  identityref

---ro values*  srlg

---ro path-srlgs-names

---ro path-srlgs-name*  [usage]

---ro usage  identityref

---ro names*  string

---ro path-route-objects

---ro path-route-object*  [index]

---ro index  uint32

---ro (type)?

---:(numbered-node-hop)

---ro numbered-node-hop

---ro node-id  te-node-id

---ro hop-type?  te-hop-type

---:(numbered-link-hop)

---ro numbered-link-hop

---ro link-tp-id  te-tp-id

---ro hop-type?  te-hop-type

---ro direction?  te-link-direction

---:(unnnumbered-link-hop)

---ro unnumbered-link-hop
```
  +--rw tunnel-termination-points
  |    +--rw source?    binary
  |    +--rw destination? binary

  +--rw tunnels
  |    +--rw sharing?   boolean
  |    +--rw tunnel* [tunnel-name]
  |       +--rw tunnel-name    string
  |       +--rw sharing?       boolean

  +--rw path-constraints
  |    +--rw te-bandwidth
  |       +--rw (technology)?
  |           +--:(generic)
  |             +--rw generic?  te-bandwidth
  |    +--rw link-protection?  identityref
  |    +--rw setup-priority?   uint8
  |    +--rw hold-priority?    uint8
  |    +--rw signaling-type?   identityref

  +--rw path-metric-bounds
  |    +--rw path-metric-bound* [metric-type]
  |       +--rw metric-type identityref
  |       +--rw upper-bound?  uint64

  +--rw path-affinities-values
  |    +--rw path-affinities-value* [usage]
  |       +--rw usage identityref
  |       +--rw value?   admin-groups

  +--rw path-affinity-names
  |    +--rw path-affinity-name* [usage]
  |       +--rw usage identityref
  |       +--rw affinity-name* [name]
  |          +--rw name    string

  +--rw path-srlgs-lists
  |    +--rw path-srlgs-list* [usage]
  |       +--rw usage identityref
  |       +--rw values*  srlg

  +--rw path-srlgs-names
  |    +--rw path-srlgs-name* [usage]
  |       +--rw usage identityref
  |       +--rw names*  string

  +--rw disjointness?
      te-path-disjointness

   +--rw optimizations
```
---rw (algorithm)?
  +--:(metric) {path-optimization-metric}?
    +--rw optimization-metric* [metric-type]
      +--rw metric-type
      |  identityref
      +--rw weight?
      |  uint8
    +--rw explicit-route-exclude-objects
      +--rw route-object-exclude-object* [index]
      +--rw index
      |  uint32
    +--rw (type)?
      +--:(numbered-node-hop)
      |  +--rw numbered-node-hop
      |      +--rw node-id
      |      +--rw hop-type?
      |      |  te-hop-type
      +--:(numbered-link-hop)
      |  +--rw numbered-link-hop
      |      +--rw link-tp-id
      |      +--rw hop-type?
      |      |  te-hop-type
      |      +--rw direction?
      |      |  te-link-direction
      +--:(unnumbered-link-hop)
      |  +--rw unnumbered-link-hop
      |      +--rw link-tp-id
      |      +--rw node-id
      |      +--rw hop-type?
      |      |  te-hop-type
      |      +--rw direction?
      |      |  te-link-direction
      +--:(as-number)
      |  +--rw as-number-hop
      |      +--rw as-number
      |      +--rw inet:as-number
+--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)
types:generalized-label
  +--ro generic?
  rt-
  +--ro direction?
    te-label-direction
  +--ro oper-status?
    te-types:te-oper-status
  +--ro geolocation
    +--ro altitude? int64
    +--ro latitude? geographic-coordinate-degree
    +--ro longitude? geographic-coordinate-degree
  +--ro statistics
    +--ro discontinuity-time? yang:date-and-time
    +--ro tunnel-termination-point
      +--ro disables? yang:counter32
      +--ro enables? yang:counter32
      +--ro maintenance-clears? yang:counter32
      +--ro maintenance-sets? yang:counter32
      +--ro modifies? yang:counter32
      +--ro downs? yang:counter32
      +--ro ups? yang:counter32
      +--ro in-service-clears? yang:counter32
      +--ro in-service-sets? yang:counter32
    +--ro local-link-connectivity
      +--ro creates? yang:counter32
      +--ro deletes? yang:counter32
      +--ro disables? yang:counter32
      +--ro enables? yang:counter32
      +--ro modifies? yang:counter32
    +--rw supporting-tunnel-termination-point*
      [node-ref tunnel-tp-ref]
      +--rw node-ref inet:uri
      +--rw tunnel-tp-ref binary
      augment /nw:networks/nw:network/nt:link:
      +--rw te!
      +--rw (bundle-stack-level)?
        +--:(bundle)
        +--rw bundled-links
          +--rw bundled-link* [sequence]
            +--rw sequence uint32
            +--rw src-tp-ref? leafref
++-rw des-tp-ref?  leafref
+-:(component)
  +-rw component-links
    +-rw component-link* [sequence]
      +-rw sequence uint32
      +-rw src-interface-ref? string
      +-rw des-interface-ref? string
  +-rw te-link-template*  
    -> ../../../../te/templates/link-template/name {template}?
  +-rw te-link-attributes
    +-rw access-type?
      te-types:te-link-access-type
    +-rw external-domain
      +-rw network-ref?  
        -> /nw:networks/network/network-id
      +-rw remote-te-node-id? te-types:te-node-id
      +-rw remote-te-link-tp-id? te-types:te-tp-id
    +-rw is-abstract?  empty
    +-rw name?  string
  +-rw underlay {te-topology-hierarchy}?
    +-rw enabled?  boolean
    +-rw primary-path
      +-rw network-ref?  
        -> /nw:networks/network/network-id
      +-rw path-element* [path-element-id]
      +-rw path-element-id uint32
      +-rw (type)?
        +-:(numbered-node-hop)
          +-rw numbered-node-hop
            +-rw node-id  te-node-id
            +-rw hop-type? te-hop-type
        +-:(numbered-link-hop)
          +-rw numbered-link-hop
            +-rw link-tp-id  te-tp-id
            +-rw hop-type? te-hop-type
            +-rw direction? te-link-direction
        +-:(unnumbered-link-hop)
          +-rw unnumbered-link-hop
            +-rw link-tp-id  te-tp-id
            +-rw node-id  te-node-id
++--ro oper-status?        te-types:te-oper-status
++--ro is-transitional?   empty
++--ro information-source? te-info-source
++--ro information-source-instance? string
++--ro information-source-state
    ++--ro credibility-preference? uint16
    ++--ro logical-network-element? string
    ++--ro network-instance? string
    ++--ro topology
        ++--ro link-ref?   leafref
++--ro information-source-entry*
        [information-source information-source-instance]
        ++--ro information-source te-info-source
        ++--ro information-source-instance string
        ++--ro information-source-state
            ++--ro credibility-preference? uint16
            ++--ro logical-network-element? string
            ++--ro network-instance? string
            ++--ro topology
                ++--ro link-ref?   leafref
                ++--ro network-ref?
                    -> /nw:networks/network/network-id
            ++--ro link-index?                       uint64
        ++--ro administrative-group?
            te-types:admin-groups
        ++--ro interface-switching-capability*
            [switching-capability encoding]
            ++--ro switching-capability identityref
            ++--ro encoding identityref
            ++--ro max-lsp-bandwidth* [priority]
                ++--ro priority   uint8
                ++--ro te-bandwidth
                    ++--ro (technology)?
                        +--:(generic)
                            ++--ro generic? te-bandwidth
            ++--ro label-restrictions
                ++--ro label-restriction* [index]
                    ++--ro restriction? enumeration
                    ++--ro index          uint32
                    ++--ro label-start
augment /nw:networks/nw:network/nw:node/nt:termination-point:
   +++rw te-tp-id?    te-types:te-tp-id
   +++rw te!
   +++rw admin-status?
      te-types:te-admin-status
   +++rw name?        string
   +++rw interface-switching-capability* [switching-capability encoding]
      +++rw switching-capability    identityref
      +++rw encoding                identityref
   +++rw max-lsp-bandwidth? [priority]
      +++rw priority        uint8
      +++rw te-bandwidth
      +++rw (technology)?
| +--rw generic?   te-bandwidth
+--rw inter-domain-plug-id?   binary
+--rw inter-layer-lock-id*   uint32
+--ro oper-status?
   |   te-types:te-oper-status
   +--ro geolocation
      +--ro altitude?   int64
      +--ro latitude?   geographic-coordinate-degree
      +--ro longitude?  geographic-coordinate-degree
Appendix B. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module ietf-te-topology defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module ietf-te-topology-state is defined as a state model, which mirrors the module ietf-te-topology defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion module, ietf-te-topology-state, is redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the module ietf-te-topology-state mirrors that of the module ietf-te-topology. The YANG tree of the module ietf-te-topology-state is not depicted separately.

B.1. TE Topology State YANG Module

This module references [RFC6001], [RFC8345], and [I-D.ietf-teas-yang-te-types].

<CODE BEGINS> file "ietf-te-topology-state@2019-02-07.yang"
module ietf-te-topology-state {
  yang-version 1.1;

  prefix "tet-s";

  import ietf-te-types {
    prefix "te-types";
    reference "I-D.ietf-teas-yang-te-types: Traffic Engineering Common YANG Types";
  }

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

  import ietf-network-state {

prefix "nw-s";
  reference "RFC 8345: A YANG Data Model for Network Topologies";
}

import ietf-network-topology-state {
  prefix "nt-s";
  reference "RFC 8345: A YANG Data Model for Network Topologies";
}

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

description
  "TE topology state model.

Copyright (c) 2019 IETF Trust and the persons identified as
authors of the code. All rights reserved."
Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info).

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.

revision "2019-02-07" {
  description "Initial revision";
  reference "RFC XXXX: YANG Data Model for TE Topologies";
  // RFC Ed.: replace XXXX with actual RFC number and remove
  // this note
}

/*
* Groupings
*/
grouping te-node-connectivity-matrix-attributes {
  description "Termination point references of a connectivity matrix entry.";
  container from {
    description "Reference to source link termination point.";
    leaf tp-ref {
      type leafref {
        path "../../../nt-s:termination-point/nt-s:tp-id";
      }
      description "Relative reference to a termination point.";
    }
    uses te-types:label-set-info;
  }
  container to {
    description "Reference to destination link termination point.";
    leaf tp-ref {
      type leafref {
        path "../../../nt-s:termination-point/nt-s:tp-id";
      }
    }
  }
}
grouping te-node-tunnel-termination-point-llc-list {
    description "Local link connectivity list of a tunnel termination point on a TE node.";
    list local-link-connectivity {
        key "link-tp-ref";
        description "The termination capabilities between tunnel-termination-point and link termination-point. The capability information can be used to compute the tunnel path. The Interface Adjustment Capability Descriptors (IACD) (defined in RFC 6001) on each link-tp can be derived from this local-link-connectivity list.";
        reference "RFC 6001: Generalized MPLS (GMPLS) Protocol Extensions for Multi-Layer and Multi-Region Networks (MLN/MRN).";
        leaf link-tp-ref {
            type leafref {
                path "../../../nt-s:termination-point/nt-s:tp-id";
            }
            description "Link termination point.";
        }
        uses te-types:label-set-info;
        uses tet:connectivity-matrix-entry-path-attributes;
    } // local-link-connectivity
} // te-node-tunnel-termination-point-config

/*
 * Data nodes
*/
augment "/nw-s:networks/nw-s:network/nw-s:network-types" {
  description
  "Introduce new network type for TE topology.";
  container te-topology {
    presence "Indicates TE topology.";
    description
    "Its presence identifies the TE topology type.";
  }
}

augment "/nw-s:networks" {
  description
  "Augmentation parameters for TE topologies.";
  uses tet:te-topologies-augment;
}

augment "/nw-s:networks/nw-s:network" {
  when "nw-s:network-types/tet-s:te-topology" {
    description
    "Augmentation parameters apply only for networks with
    TE topology type.";
  }
  description
  "Configuration parameters for TE topology.";
  uses tet:te-topology-augment;
}

augment "/nw-s:networks/nw-s:network/nw-s:node" {
  when ".../nw-s:network-types/tet-s:te-topology" {
    description
    "Augmentation parameters apply only for networks with
    TE topology type.";
  }
  description
  "Configuration parameters for TE at node level.";
  leaf te-node-id {
    type te-types:te-node-id;
    description
    "The identifier of a node in the TE topology.
    A node is specific to a topology to which it belongs.";
  }
}
container te {
  must "../te-node-id" {
    description
    "te-node-id is mandatory.";
  }
  must "count(../nw-s: supporting-node) <= 1" {
    description
    "For a node in a TE topology, there cannot be more
    than 1 supporting node. If multiple nodes are abstracted,
    the underlay-topology is used.";
  }
  presence "TE support.";
  description
  "Indicates TE support.";
  uses tet:te-node-augment;
} // te

augment "/nw-s:networks/nw-s: network/nt-s:link" {
  when ".../nw-s: network-types/tet-s:te-topology" {
    description
    "Augmentation parameters apply only for networks with
    TE topology type.";
  }
  description
  "Configuration parameters for TE at link level.";
  container te {
    must "count(../nt-s: supporting-link) <= 1" {
      description
      "For a link in a TE topology, there cannot be more
      than 1 supporting link. If one or more link paths are
      abstracted, the underlay is used.";
    }
    presence "TE support.";
    description
    "Indicates TE support.";
    uses tet:te-link-augment;
  } // te
}
augment "/nw-s:networks/nw-s:network/nw-s:node/
+ "nt-s:termination-point" {
    when "../../nw-s:network-types/tet-s:te-topology" {
        description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
    "Configuration parameters for TE at termination point level.";
    uses tet:te-termination-point-augment;
}

+ "bundle/bundled-links/bundled-link" {
    when "../../../../nw-s:network-types/tet-s:te-topology" {
        description
        "Augmentation parameters apply only for networks with
        TE topology type.";
    }
    description
    "Augment TE link bundled link.";
    leaf src-tp-ref {
        type leafref {
            path "../../../nw-s:node[nw-s:node-id = "
            + "current()//.../.../nt-s:source/
            + "nt-s:source-node"]/
            + "nt-s:termination-point/nt-s:tp-id";
            require-instance true;
        }
        description
        "Reference to another TE termination point on the
        same source node.";
    }
    leaf des-tp-ref {
        type leafref {
            path "../../../nw-s:node[nw-s:node-id = "
            + "current()//.../.../nt-s:destination/
            + "nt-s:dest-node"]/
            + "nt-s:termination-point/nt-s:tp-id";
            require-instance true;
        }
    }
description
  "Reference to another TE termination point on the
  same destination node.";
}
}

augment
"/nw-s:networks/nw-s:network/nw-s:node/te/
+ "information-source-entry/connectivity-matrices/
+ "connectivity-matrix" {
  when "../../../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
"/nw-s:networks/nw-s:network/nw-s:node/te/te-node-attributes/
+ "connectivity-matrices/connectivity-matrix" {
  when "../../../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}

augment
"/nw-s:networks/nw-s:network/nw-s:node/te/
+ "tunnel-termination-point/local-link-connectivities" {
  when "../../../nw-s:network-types/tet-s:te-topology" {
    description
      "Augmentation parameters apply only for networks with
      TE topology type.";
  }
  description
    "Augment TE node connectivity-matrix.";
  uses te-node-connectivity-matrix-attributes;
}  
  description  
    "Augment TE node tunnel termination point LLCs  
    (Local Link Connectivities).";  
    uses te-node-tunnel-termination-point-llc-list;  
};  
}  

<CODE ENDS>
Appendix C. Example: YANG Model for Technology Specific Augmentations

This section provides an example YANG module to define a technology specific TE topology model for the example-topology described in Section 6.

module example-topology {
  yang-version 1.1;

  namespace "http://example.com/example-topology";
  prefix "ex-topo";

  import ietf-network {
    prefix "nw";
  }

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

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

  organization
    "Example Organization";
  contact
    "Editor: Example Author";

  description
    "This module defines a topology data model for the example technology.";

  revision 2018-06-15 {
    description
      "Initial revision."
    reference
      "Example reference."
  }

  /*
   * Data nodes
* /
augment "/nw:networks/nw:network/nw:network-types/"
+ "tet:te-topology" {
  description
  "Augment network types to define example topology type.";
  container example-topology {
    presence
    "Introduce new network type for example topology.";
    description
    "Its presence identifies the example topology type.";
  }
}

augment "/nw:networks/nw:network/tet:te" {
  when "/nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology/" {
    description
    "Augmentation parameters apply only for networks with
    example topology type.";
  }
  description "Augment network topology.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-1 {
      type uint8;
      description "Attribute 1 for example technology.";
    }
  }
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes" {
  when "/nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology/" {
    description
    "Augmentation parameters apply only for networks with
    example topology type.";
  }
  description "Augment node attributes.";
  container attributes {
    description "Attributes for example technology.";
  }
}
leaf attribute-2 {
  type uint8;
  description "Attribute 2 for example technology.";
}
}
augment "/nw:networks/nw:network/nw:node/tet:te/" 
+ "tet:te-node-attributes/tet:connectivity-matrices" {
  when "./././././nw:network-types/tet:te-topology/" 
+ "ex-topo:example-topology" {
    description 
    "Augmentation parameters apply only for networks with 
    example topology type.";
  }
  description "Augment node connectivity matrices.";
  container attributes {
    leaf attribute-3 {
      type uint8;
      description "Attribute 3 for example technology.";
    }
  }
}
}
augment "/nw:networks/nw:network/nw:node/tet:te/" 
+ "tet:te-node-attributes/tet:connectivity-matrices/" 
+ "tet:connectivity-matrix" {
  when "./././././nw:network-types/tet:te-topology/" 
+ "ex-topo:example-topology" {
    description 
    "Augmentation parameters apply only for networks with 
    example topology type.";
  }
  description "Augment node connectivity matrix.";
  container attributes {
    description "Attributes for example technology.";
    leaf attribute-3 {
      type uint8;
      description "Attribute 3 for example technology.";
    }
  }
}
augment "/nw:networks/nw:network/nw:node/tet:te/"
    + "tet:tunnel-termination-point" {
when "../../../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
    }
    description "Augment tunnel termination point.";
    container attributes {
        description "Attributes for example technology.";
        leaf attribute-4 {
            type uint8;
            description "Attribute 4 for example technology.";
        }
    }
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/"
    + "tet:te" {
when "../../../nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
    description
        "Augmentation parameters apply only for networks with
        example topology type.";
    }
    description "Augment link termination point.";
    container attributes {
        description "Attributes for example technology.";
        leaf attribute-5 {
            type uint8;
            description "Attribute 5 for example technology.";
        }
    }
}

    + "tet:te-link-attributes" {
when "../../../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
       example topology type.";
  }
description "Augment link attributes.";
container attributes {
  description "Attributes for example technology.";
  leaf attribute-6 {
    type uint8;
    description "Attribute 6 for example technology.";
  }
}
}
/*
 * Augment TE bandwidth.
 */
augment "../nw:networks/tet:te/tet:templates/
  + tet:link-template/tet:te-link-attributes/
  + tet:interface-switching-capability/tet:max-lsp-bandwidth/
  + tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
}description "Augment TE bandwidth.";
}
augment "../nw:networks/tet:te/tet:templates/
  + tet:link-template/tet:te-link-attributes/
  + tet:max-link-bandwidth/
  + tet:te-bandwidth/tet:technology" {
  case "example" {

container example {
  description "Attributes for example technology.";
  leaf bandwidth-1 {
    type uint32;
    description "Bandwidth 1 for example technology.";
  }
}

description "Augment TE bandwidth.";
}

augment "/nw:networks/tet:te/tet:templates/"
  + "tet:link-template/tet:te-link-attributes/"
  + "tet:max-resv-link-bandwidth/"
  + "tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
}

description "Augment TE bandwidth.";
}

augment "/nw:networks/tet:te/tet:templates/"
  + "tet:link-template/tet:te-link-attributes/"
  + "tet:unreserved-bandwidth/"
  + "tet:te-bandwidth/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology.";
      }
    }
  }
}

description "Augment TE bandwidth.";
augment "/nw:networks/nw:network/nw:node/tet:te/" 
+ "tet:te-node-attributes/tet:connectivity-matrices/" 
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" 
when "/nw:networks/nw:network/nw:node/tet:te/" 
+ "tet:te-node-attributes/tet:connectivity-matrices/" 
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" 
when "ex-topo:example-topology" { 

description 
"Augmentation parameters apply only for networks with example topology type.";
}
case "example" { 

case "example" { 

description "Augment TE bandwidth.";
}
}
+ "tet:information-source-entry/tet:connectivity-matrices/
+ "ex-topo:example-topology" { description "Augmentation parameters apply only for networks with example topology type.";
} case "example" {
  container example {
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
} description "Augment TE bandwidth.";
}
  + "tet:tunnel-termination-point/tet:client-layer-adaptation/
  + "tet:switching-capability/tet:te-bandwidth/tet:technology"
  { when "../../../../../../../nw:network-types/tet:te-topology/
    + "ex-topo:example-topology" { description
      "Augmentation parameters apply only for networks with
      example topology type.";
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf bandwidth-1 {
          type uint32;
          description "Bandwidth 1 for example technology.";
        }
      }
    }
  }
  description "Augment TE bandwidth.";
}
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/"
+ "tet:path-constraints/tet:te-bandwidth/tet:technology" 
  when "./././././././././nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" 
  description 
  "Augmentation parameters apply only for networks with 
  example topology type.";
}
  case "example" { 
    container example { 
      description "Attributes for example technology.";
      leaf bandwidth-1 { 
        type uint32;
        description "Bandwidth 1 for example technology.";
      } 
    } 
  } 
  description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
+ "tet:te-link-attributes/
+ "tet:interface-switching-capability/tet:max-lsp-bandwidth/
+ "tet:te-bandwidth/tet:technology" 
  when "./././././././././nw:network-types/tet:te-topology/"
+ "ex-topo:example-topology" 
  description 
  "Augmentation parameters apply only for networks with 
  example topology type.";
}
  case "example" { 
    container example { 

Liu, et al Expires December 19, 2019 [Page 181]
leaf bandwidth-1 {
    type uint32;
    description "Bandwidth 1 for example technology."
}
}

description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:te-link-attributes/
  + "tet:max-link-bandwidth/
  + "tet:te-bandwidth/tet:technology" {
when "../../../../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
    "Augmentation parameters apply only for networks with example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology."
      }
    }
  }
  description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:te-link-attributes/
  + "tet:max-resv-link-bandwidth/
  + "tet:te-bandwidth/tet:technology" {
when "../../../../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description
    "Augmentation parameters apply only for networks with example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf bandwidth-1 {
        type uint32;
        description "Bandwidth 1 for example technology."
      }
    }
  }
  description "Augment TE bandwidth.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
}
description "Augmentation parameters apply only for networks with example topology type.";
}
} case "example" {
  container example {
    description "Attributes for example technology.";
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
}
}

description "Augment TE bandwidth.";
}

  + "tet:information-source-entry/
  + "tet:max-resv-link-bandwidth/
  + "tet:te-bandwidth/tet:technology" {
when "..//..//..//..//nw:network-types/tet:te-topology/
  + "ex-topo:example-topology" {
    description "Augmentation parameters apply only for networks with example topology type.";
}
} case "example" {
  container example {
    description "Attributes for example technology.";
    leaf bandwidth-1 {
      type uint32;
      description "Bandwidth 1 for example technology.";
    }
  }
}
}

description "Augment TE bandwidth.";
}

  + "tet:information-source-entry/
  + "tet:unreserved-bandwidth/"
+ "tet:te-bandwidth/tet:technology" {
    when "./..././..././nw:network-types/tet:te-topology/
        + "ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf bandwidth-1 {  
                type uint32;
                description "Bandwidth 1 for example technology.";
            }
        }
    }
    description "Augment TE bandwidth.";
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/
    + "tet:te/
    + "tet:interface-switching-capability/tet:max-lsp-bandwidth/
    + "tet:te-bandwidth/tet:technology" {
    when "./..././..././..././nw:network-types/tet:te-topology/
        + "ex-topo:example-topology" {
        description
            "Augmentation parameters apply only for networks with
            example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf bandwidth-1 {  
                type uint32;
                description "Bandwidth 1 for example technology.";
            }
        }
    }
    description "Augment TE bandwidth.";
}
/* *
* Augment TE label. *
*/

+ "tet:link-template/tet:te-link-attributes/
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

+ "tet:link-template/tet:te-link-attributes/
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

+ "tet:link-template/tet:te-link-attributes/
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/
+ "tet:te-label/tet:technology" {
  case "example" {
    
Liu, et al Expires December 19, 2019 [Page 186]
container example {
  description "Attributes for example technology.";
  leaf label-1 {
    type uint32;
    description "Label 1 for example technology.";
  }
}

description "Augment TE label.";

augment "/nw:networks/tet:te/tet:templates/"
+ "tet:link-template/tet:te-link-attributes/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}

/* Under te-node-attributes/connectivity-matrices */

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  when "...../....../....../....../nw:network-types/tet:te-topology/"
  + "ex-topo:example-topology" {
    description "Augmentation parameters apply only for networks with example topology type.";
  }
  case "example" {
    container example {

description "Attributes for example technology.";
leaf label-1 {
  type uint32;
  description "Label 1 for example technology.";
}
}
description "Augment TE label.";
}

  + "tet:te-node-attributes/tet:connectivity-matrices/
  + "tet:label-restrictions/tet:label-restriction/tet:label-end/"
  + "tet:te-label/tet:technology" {
    when "/.\.\.\.\.\.\.\.\.\.\.\.\.\nw:network-types/tet:te-topology/"
    + "ex-topo:example-topology" {
      description
      "Augmentation parameters apply only for networks with
      example topology type.";
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
      description "Augment TE label.";
    }
}}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}

description "Augment TE label.";

}
+ "tet:te-topology/ex-topo:example-topology" {
    description
    "Augmentation parameters apply only for networks with
example topology type.";
}

case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}

description "Augment TE label.";

/* Under te-node-attributes/.../connectivity-matrix */
+ "tet:te-node-attributes/tet:connectivity-matrices/
+ "tet:connectivity-matrix/tet:from/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/
+ "tet:te-label/tet:technology" {
    when "/nw:network-types/
+ "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
example topology type.";
    }

case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}

description "Augment TE label.";
augment "*/nw:networks/nw:network/nw:node/tet:te/"
  + "tet:te-node-attributes/tet:connectivity-matrices/"
  + "tet:connectivity-matrix/tet:from/"
  + "tet:label-restrictions/tet:label-restriction/tet:label-end/"
  + "tet:te-label/tet:technology" {
    when "....../....../....../....../....../nw:network-types/"
      + "tet:te-topology/ex-topo:example-topology" {
        description
          "Augmentation parameters apply only for networks with
          example topology type.";
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
    description "Augment TE label.";
  }

augment "*/nw:networks/nw:network/nw:node/tet:te/"
  + "tet:te-node-attributes/tet:connectivity-matrices/"
  + "tet:connectivity-matrix/tet:to/"
  + "tet:label-restrictions/tet:label-restriction/tet:label-start/"
  + "tet:te-label/tet:technology" {
    when "....../....../....../....../....../nw:network-types/"
      + "tet:te-topology/ex-topo:example-topology" {
        description
          "Augmentation parameters apply only for networks with
          example topology type.";
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
    description "Augment TE label.";
augment "/nw:networks/nw:network/nw:node/tet:te/"
  + "tet:te-node-attributes/tet:connectivity-matrices/"
  + "tet:connectivity-matrix/tet:to/"
  + "tet:label-restrictions/tet:label-restriction/tet:label-end/"
  + "tet:te-label/tet:technology" {
    when "../../../../../../../../nw:network-types/"
      + "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
         example topology type.";
    }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
container example {
    description "Attributes for example technology.";
    leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
    }
}
}
description "Augment TE label.";
}

+ /tet:te-node-attributes/tet:connectivity-matrices/
+ /tet:connectivity-matrix/
+ /tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ".../.../.../.../.../.../.../.../.../nw:network-types/
+ /tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
example topology type.";
    }
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
}
description "Augment TE label.";
}

+ /tet:te-node-attributes/tet:connectivity-matrices/
+ /tet:connectivity-matrix/
+ /tet:path-properties/tet:path-route-objects/
+ /tet:path-route-object/tet:type/
+ /tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ".../.../.../.../.../.../.../.../.../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
  "Augmentation parameters apply only for networks with
  example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}

/* Under information-source-entry/connectivity-matrices */
+ "tet:information-source-entry/tet:connectivity-matrices/
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/
+ "tet:te-label/tet:technology" {
  when "/nw:network-types/tet:te-topology/
+ "ex-topo:example-topology" {
    description
    "Augmentation parameters apply only for networks with
    example topology type.";
  }
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}
augment "'/nw:networks/nw:network/nw:node/tet:te/
 + 'tet:information-source-entry/tet:connectivity-matrices/
 + 'tet:label-restrictions/tet:label-restriction/tet:label-end/
 + 'tet:te-label/tet:technology" {
 when "../../../../nw:network-types/tet:te-topology/
 + 'ex-topo:example-topology" {
 description
 "Augmentation parameters apply only for networks with
 example topology type.";
 }
 case "example" {
 container example {
   description "Attributes for example technology.";
   leaf label-1 {
     type uint32;
     description "Label 1 for example technology.";
   }
 }
 description "Augment TE label.";
 }
 }

augment "'/nw:networks/nw:network/nw:node/tet:te/
 + 'tet:information-source-entry/tet:connectivity-matrices/
 + 'tet:underlay/tet:primary-path/tet:path-element/tet:type/
 + 'tet:label/tet:label-hop/tet:te-label/tet:technology" {
 when "../../../../nw:network-types/tet:te-topology/
 + 'ex-topo:example-topology" {
 description
 "Augmentation parameters apply only for networks with
 example topology type.";
 }
 case "example" {
 container example {
   description "Attributes for example technology.";
   leaf label-1 {
     type uint32;
     description "Label 1 for example technology.";
   }
 }
}
description "Augment TE label.";

augment "/*/nw:networks/nw:network/nw:node/tet:te/
  + "tet:information-source-entry/tet:connectivity-matrices/
  + "tet:underlay/tet:backup-path/tet:path-element/tet:type/
  + "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ""/nw:network-types/tet:te-topology/ex-topo:example-topology"
    + "tet:te-topology/ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
    }
  }
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";

augment "/*/nw:networks/nw:network/nw:node/tet:te/
  + "tet:information-source-entry/tet:connectivity-matrices/
  + "tet:path-properties/tet:path-route-objects/
  + "tet:path-route-object/tet:type/
  + "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ""/nw:network-types/tet:te-topology/ex-topo:example-topology"
    + "tet:te-topology/ex-topo:example-topology" {
    description
      "Augmentation parameters apply only for networks with
      example topology type.";
    }
  }
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
augment "/nw:networks/nw:network/nw:node/tet:te/"
  + "tet:information-source-entry/tet:connectivity-matrices/"
  + "tet:connectivity-matrix/tet:from/"
  + "tet:label-restrictions/tet:label-restriction/tet:label-start/"
  + "tet:te-label/tet:technology" {
    when "../../../../../../../../nw:network-types/
      + "tet:te-topology/ex-topo:example-topology" {
      description "Augmentation parameters apply only for networks with
                    example topology type.";
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
  }
  description "Augment TE label.";
}
example topology type.
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}

description "Augment TE label.
"
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  when "../../../../../../../../nw:network-types/"
  + "tet:te-topology/ex-topo:example-topology" {
    description
    "Augmentation parameters apply only for networks with
example topology type.";
  }
  }
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}

description "Augment TE label.
"
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix/tet:to/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
    when "../../../../../../../../nw:network-types/
+ "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
         example topology type.";
    }
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}

+ "tet:information-source-entry/tet:connectivity-matrices/
+ "tet:connectivity-matrix/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when "../../../../../../../../nw:network-types/
+ "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
         example topology type.";
    }
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}
description "Augment TE label.";
}
  + "tet:information-source-entry/tet:connectivity-matrices/
  + "tet:connectivity-matrix/"
  + "tet:underlay/tet:backup-path/tet:path-element/tet:type/
  + "tet:path-properties/tet:path-route-objects/
  + "tet:path-route-object/tet:type/
  + "tet:te-topology/ex-topo:example-topology" { description
  "Augmentation parameters apply only for networks with example topology type.";
} case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
} description "Augment TE label.";
}

  + "tet:information-source-entry/tet:connectivity-matrices/
  + "tet:connectivity-matrix/"
  + "tet:path-properties/tet:path-route-objects/
  + "tet:path-route-object/tet:type/
  + "tet:te-topology/ex-topo:example-topology" { description
  "Augmentation parameters apply only for networks with example topology type.";
} case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
    }
  }
} description "Augment TE label.";
description "Label 1 for example technology.";
}
}
)
description "Augment TE label.";
}
/* Under tunnel-termination-point/local-link-connectivities */
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/"
+ "tet:te-label/tet:technology" {
  when "/.../.../.../.../.../nw:network-types/tet:te-topology/
+ "ex-topo:example-topology" {
    description "Augmentation parameters apply only for networks with
example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label.";
}
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/"
+ "tet:te-label/tet:technology" {
  when "/.../.../.../.../.../nw:network-types/tet:te-topology/
+ "ex-topo:example-topology" {
    description "Augmentation parameters apply only for networks with
example topology type.";
  }
}
case "example" {
    container example {
        description "Attributes for example technology.";
        leaf label-1 {
            type uint32;
            description "Label 1 for example technology.";
        }
    }
}

description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ""..."/.../..."/.../.../.../"nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
         example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
}

description "Augment TE label.";
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when ""..."/.../..."/.../.../.../"nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
        description
        "Augmentation parameters apply only for networks with
         example topology type.";
    }
    case "example" {
        container example {
            description "Attributes for example technology.";
            leaf label-1 {
                type uint32;
                description "Label 1 for example technology.";
            }
        }
    }
}

"Augmentation parameters apply only for networks with example topology type."
}
case "example" {
  container example {
    description "Attributes for example technology."
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology."
    }
  }
}

description "Augment TE label.";
}
  + "tet:tunnel-termination-point/tet:local-link-connectivities/
  + "tet:path-properties/tet:path-route-objects/
  + "tet:path-route-object/tet:type/
  + "tet:label/tet:label-hop/tet:te-label/tet:technology" {
  when "././././././././././././nw:network-types/
    + "tet:te-topology/ex-topo:example-topology" {
    description "Augmentation parameters apply only for networks with example topology type."
  }
}
case "example" {
  container example {
    description "Attributes for example technology."
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology."
    }
  }
}

description "Augment TE label.";
}

/* Under tunnel-termination-point/.../local-link-connectivity */
augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/
+ "tet:label-restrictions/tet:label-restriction/tet:label-start/
+ "tet:te-label/tet:technology" {
when "../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
  "Augmentation parameters apply only for networks with
  example topology type.";
}

case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}

description "Augment TE label.";
}

+ "tet:tunnel-termination-point/tet:local-link-connectivities/"
+ "tet:local-link-connectivity/
+ "tet:label-restrictions/tet:label-restriction/tet:label-end/
+ "tet:te-label/tet:technology" {
when "../../../nw:network-types/"
+ "tet:te-topology/ex-topo:example-topology" {
  description
  "Augmentation parameters apply only for networks with
  example topology type.";
}
}

case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label."
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
+ "tet:tunnel-termination-point/tet:local-link-connectivities/
+ "tet:local-link-connectivity/
+ "tet:underlay/tet:primary-path/tet:path-element/tet:type/
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" { when
+ "nw:network-types/
+ "tet:te-topology/ex-topo:example-topology" {
  description
    "Augmentation parameters apply only for networks with
    example topology type.";
  }
  case "example" {
    container example {
      description "Attributes for example technology.";
      leaf label-1 {
        type uint32;
        description "Label 1 for example technology.";
      }
    }
  }
  description "Augment TE label."
}
leaf label-1 {
  type uint32;
  description "Label 1 for example technology."
}

augment "/nw:networks/nw:network/nw:node/tet:te/"
  + "tet:tunnel-termination-point/tet:local-link-connectivities/"
  + "tet:local-link-connectivity/"
  + "tet:path-properties/tet:path-route-objects/"
  + "tet:path-route-object/tet:type/"
  + "tet:label/tet:label-hop/tet:te-label/tet:technology" {
    when "/nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {
      description "Augmentation parameters apply only for networks with
                     example topology type."
    }
    case "example" {
      container example {
        description "Attributes for example technology."
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology."
        }
      }
    }
    description "Augment TE label.";
  }

/* Under te-link-attributes */

augment "/nw:networks/nw:network/nt:link/tet:te/"
  + "tet:te-link-attributes/"
  + "tet:label-restrictions/tet:label-restriction/tet:label-start/"
  + "tet:te-label/tet:technology" {
    when "/nw:network-types/"
    + "tet:te-topology/ex-topo:example-topology" {

Liu, et al Expires December 19, 2019 [Page 206]
description
"Augmentation parameters apply only for networks with
example topology type.";
}
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}
  + "tet:te-link-attributes/
  + "tet:label-restrictions/tet:label-restriction/tet:label-end/"
  + "tet:te-label/tet:technology" {
    when "..../..../..../..../..../..../nw:network-types/
      + "tet:te-topology/ex-topo:example-topology" {
      description "Augmentation parameters apply only for networks with
example topology type.";
    }
  }
case "example" {
  container example {
    description "Attributes for example technology.";
    leaf label-1 {
      type uint32;
      description "Label 1 for example technology.";
    }
  }
}
description "Augment TE label.";
}
  + "tet:te-link-attributes/
  + "tet:underlay/tet:primary-path/tet:path-element/tet:type/"
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" { 
  when "/nw:network-types/" 
  + "tet:te-topology/ex-topo:example-topology" { 
    description 
    "Augmentation parameters apply only for networks with 
    example topology type."; 
  } 
  case "example" { 
    container example { 
      description "Attributes for example technology."; 
      leaf label-1 { 
        type uint32; 
        description "Label 1 for example technology."; 
      } 
    } 
  } 
  description "Augment TE label."; 
} 

augment "/nw:networks/nw:network/nt:link/tet:te/" 
+ "tet:link-attributes/" 
+ "tet:underlay/tet:backup-path/tet:path-element/tet:type/" 
+ "tet:label/tet:label-hop/tet:te-label/tet:technology" { 
  when "/nw:network-types/" 
  + "tet:te-topology/ex-topo:example-topology" { 
    description 
    "Augmentation parameters apply only for networks with 
    example topology type."; 
  } 
  case "example" { 
    container example { 
      description "Attributes for example technology."; 
      leaf label-1 { 
        type uint32; 
        description "Label 1 for example technology."; 
      } 
    } 
  } 
  description "Augment TE label."; 
}
/* Under te-link information-source-entry */

  + "tet:information-source-entry/
  + "tet:label-restrictions/tet:label-restriction/tet:label-start/
  + "tet:te-label/tet:technology" {
    when "../../../../../../../nw:network-types/
      + "tet:te-topology/ex-topo:example-topology" {
      description
      "Augmentation parameters apply only for networks with
      example topology type."
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
    description "Augment TE label.";
  }

  + "tet:information-source-entry/
  + "tet:label-restrictions/tet:label-restriction/tet:label-end/
  + "tet:te-label/tet:technology" {
    when "../../../../../../../nw:network-types/
      + "tet:te-topology/ex-topo:example-topology" {
      description
      "Augmentation parameters apply only for networks with
      example topology type."
    }
    case "example" {
      container example {
        description "Attributes for example technology.";
        leaf label-1 {
          type uint32;
          description "Label 1 for example technology.";
        }
      }
    }
    description "Augment TE label.";
Contributors

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

Dieter Beller
Nokia
Email: Dieter.Beller@nokia.com

Carlo Perocchio
Ericsson
Email: carlo.perocchio@ericsson.com

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

Authors’ Addresses

Xufeng Liu
Volta Networks
Email: xufeng.liu.ietf@gmail.com

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

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

Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

Himanshu Shah
Ciena
Email: hshah@ciena.com
GMPES Routing and Signaling Framework for Flexible Ethernet (FlexE)
draft-izh-ccamp-flexe-fwk-00

Abstract

Traditionally, Ethernet MAC rates were constrained to match the rates of the Ethernet PHY(s). OIF’s implementation agreement [OIFMLG3] was the first step in allowing MAC rates to be different than the PHY rates. OIF has recently approved another implementation agreement [OIFFLEXE1] which allows complete decoupling of the MAC data rates and the Ethernet PHY(s) that support them. This includes support for (a) MAC rates which are greater than the rate of a single PHY (satisfied by bonding of multiple PHY(s)), (b) MAC rates which are less than the rate of a PHY (sub-rate), (c) support of multiple FlexE client signals carried over a single PHY, or over a collection of bonded PHY(s). The FlexE SHIM functions which bond mulitple Ethernet PHY(s) to form a large "pipe" view the connectivity between two FlexE aware devices as a collection of multiple point-to-point links (one link per Ethernet PHY). These logical point-to-point links can either be direct links (without an intervening transport network), or realized via a Optical transport network. This draft catalogs the usecases that capture the FlexE deployment scenarios -- including the cases that include/exclude OTNs.

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 23, 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. Requirements Language ............................... 4
2. Terminology ............................................. 4
3. Usecases ................................................ 5
   3.1. FlexE unaware transport ............................. 5
   3.2. FlexE Aware .......................................... 7
       3.2.1. FlexE Aware Case - No Resizing .................. 7
   3.3. FlexE Termination - Transport ........................ 11
       3.3.1. FlexE Client at Both endpoints ................ 11
       3.3.2. Interworking of FlexE Client w/ Native Client at the other endpoint .......................... 12
       3.3.3. Interworking of FlexE client w/ Client from OIF_MLG ........................................ 14
       3.3.4. Back-to-Back FlexE ................................. 15
           3.3.4.1. FlexE Client BW Resizing ..................... 15
4. Requirements ............................................. 16
5. Framework ................................................ 17
6. Architecture ............................................. 17
7. Solution ................................................ 17
8. Acknowledgements ........................................ 17
Traditionally, Ethernet MAC rates were constrained to match the rates of the Ethernet PHY(s). OIF’s implementation agreement [OIFMLG3] was the first step in allowing MAC rates to be different than the PHY rates standardized by IEEE. OIF has recently approved another implementation agreement [OIFFLEXE1] which allows complete decoupling of the MAC data rates and the Ethernet PHY(s) that support them. This includes support for (a) MAC rates which are greater than the rate of a single PHY (satisfied by bonding of multiple PHY(s)), (b) MAC rates which are less than the rate of a PHY (sub-rate), (c) support of multiple FlexE client signals carried over a single PHY, or over a collection of bonded PHY(s). The capabilities supported by the OIF FlexE implementation agreement version 1.0 are:

a. Support a large rate Ethernet MAC over bonded Ethernet PHYs, e.g. supporting a 200G MAC over 2 bonded 100GBASE-R PHY(s)

b. Support a sub-rate Ethernet MAC over a single Ethernet PHY, e.g. supporting a 50G MAC over a 100GBASE-R PHY

c. Support a collection of flexible Ethernet clients over a single Ethernet PHY, e.g. supporting two MACs with the rates 25G, 50G over a single 100GBASE-R PHY

d. Support a sub-rate Ethernet MAC over bonded PHYs, e.g. supporting a 150G Ethernet client over 2 bonded 100GBASE-R PHY(s)

e. Support a collection of Ethernet MAC clients over bonded Ethernet PHYs, e.g. supporting a 50G, and 150G MAC over 2 bonded Ethernet PHY(s)

All networks which support the bonding of Ethernet interfaces (as per [OIFFLEXE1]) include a basic building block -- which consists of two FlexE SHIM functions (located at opposite ends of a link) and the (logical) point to point links that carry the Ethernet PHY signals between the two FlexE SHIM Functions. These logical point-to-point PHY links can be realized in a variety of ways:
a. These are direct point-to-point links with no intervening transport network.

b. The Ethernet PHY(s) are transparently transported via an Optical Transport Network. Optical Transport Networks (defined by [G709] and [G798]) have recently expanded the traditional bit (or codeword) transparent transport of Ethernet client signals, and included support for the use cases identified in the OIF FlexE implementation agreement.

c. Realized by tunneling the Ethernet PHY(s) over some other type of network (e.g. IP/MPLS). Thus, for example, the Ethernet PHY(s) signals could be carried over a pseudowire (or a LSP) in the IP/MPLS network. Note that the OIF implementation agreement [OIFFLEXE1] only includes support for 100G Ethernet PHY(s). As a result of this encapsulation into a PW, the bandwidth of the PW will be much larger than the bit rate of the Ethernet PHY (i.e. 100G), and such a pseudowire cannot be transported in networks that only include 100G Ethernet links. This scenario is realizable when (a) higher rate Ethernet PHY(s), e.g. 200G/40G are supported) or (b) OIF extends the FlexE groups to include lower rate Ethernet PHY(s), e.g. at the 25G/50G rate. Further study is needed to ensure that these scenarios are realizable, practical, and beneficial to operators. With this in mind, the current draft doesn’t include any coverage for this scenario.

Internet-draft examines the use cases that arise when the logical links between FlexE capable devices are (a) point-to-point links without any intervening network (b) realized via Optical transport networks. This draft considers the variants in which the two peer FlexE devices are both customer-edge devices, or customer-edge/provider edge devices. This list of use cases will help identify the Control Plane (i.e. Routing and Signaling) extensions that may be required).

1.1. Requirements Language

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

2. Terminology

a. Ethernet PHY: an entity representing 100G-R Physical Coding Sublayer (PCS), Physical Media Attachment (PMA), and Physical Media Dependent (PMD) layers.
b. FlexE Group: A FlexE Group is composed of from 1 to n 100GBASE-R Ethernet PHYs. Each PHY is identified by a number in the range [1-254].

c. FlexE Client: an Ethernet flow based on a MAC data rate that may or may not correspond to any Ethernet PHY rate (e.g., 10, 40, m x 25 Gb/s).

d. FlexE Shim: the layer that maps or demaps the FlexE clients carried over a FlexE group.

e. FlexE Calendar: The total capacity of a FlexE group is represented as a collection of slots which have a granularity of 5G. The calendar for a FlexE group composed of n 100G PHYs is represented as an array of 20n slots (each representing 5G of bandwidth). This calendar is partitioned into sub-calendars, with 20 slots per 100G PHY. Each FlexE client is mapped into one or more calendar slots (based on the bandwidth of the FlexE client).

3. Use cases

3.1. FlexE unware transport

The FlexE shim layer in a router maps the FlexE client(s) over the FlexE group. The transport network is unaware of the FlexE. Each of the FlexE group PHY is carried independently across the transport network over the same fiber route. The FlexE shim in the router tolerates end-to-end skew across the network. In this use case, the router makes flexible use of the full capacity of the FlexE group, and depends on legacy transport equipment to realize PCS-codeword-transparent transport of 100GbE. It allows striping of PHYs in the FlexE group over multiple line cards in the transport equipment. It is worth mentioning that in this case, the FlexE SHIM layer is terminated at the routers, and the coordination of operations related to FlexE clients, e.g. creating new FlexE clients, deleting existing FlexE clients, and resizing the bandwidth of existing FlexE clients (if desired) happens between the two routers. Note that the transport network is completely transparent to the FlexE signals, and doesn’t participate in any FlexE protocols.
Figure 1: FlexE unaware transport
3.2. FlexE Aware

3.2.1. FlexE Aware Case - No Resizing

This scenario represents an optimization of the FlexE unaware transport presented in Section 3.1, and illustrated in Figure 1. In this application (see Figure 2), the devices at the edge of the transport network do not terminate the FlexE shim layer, but are aware of the (a) composition of the FlexE group (i.e. set of all contained Ethernet PHYs) and (b) format of the FlexE overhead. They "snoop" the FlexE overhead to determine the subset of the set of all calendar slots that are available for use (i.e. these calendar slots may be used, or unused). The transport network edge removes the unavailable calendar slots at the ingress to the network, and adds the same unavailable calendar slots back when exiting the network. The result is that the FlexE Shim layers at both routers see exactly the same input that they saw in the FlexE unaware scenario -- with the added benefit that the line (or DWDM) side bandwidth has been optimized to be sufficient to carry only the available calendar slots in all of the Ethernet PHY(s) in the FlexE group. This mode may be used in cases where the bandwidth of the Ethernet PHY is greater than the bit rate supported by a wavelength (and it is known that that all calendar slots in the PHY are not "available").

The transport network edge device could learn of the set of unavailable calendar slots in a variety of ways; a few examples are listed below:

a. The set of unavailable calendar slots could be configured against each Ethernet PHY in the FlexE group. The FlexE demux function in the transport network edge device (A) compares the information about calendar slots which are expected to be unavailable (as per user supplied configuration), with the corresponding information encoded by the customer edge device in the FlexE overhead (as specified in [OIFFLEXE1]). If there is a mismatch between the unavailable calendar slots in any of the PHYs within a FlexE group, the transport edge node software could raise an alarm to report the inconsistency between the provisioning information at the transport network edge, and the customer edge device.

b. The Transport network edge could be configured to act in a "slave" mode. In this mode, the FlexE demux function at the Transport network edge (A) receives the information about the available/unavailable calendar slots by observing the FlexE overhead (as specified in [OIFFLEXE1]) and uses this information to select (a) the set of wavelengths (with appropriate capacities) or (b) the bandwidth of the ODUflex (or fixed rate ODUks) that could carry the FlexE PCS end-to-end.
c. The set of unavailable slots could be negotiated between FlexE Shim entity in the customer device and the partial rate ODUflex mapper located in the transport network element. Thus, for example, the transport network element could declare the maximum number of 5G slots that could be transported over a single wavelength, and the customer network device can choose the number of 5G slots that will be used between customer devices. This process could be accomplished through control protocols such as LMP, using the appropriate control channel for transporting the messages.

In the basic FlexE aware mode, the transport network edge does not expect the number of unavailable calendar slots to change dynamically.

Note that the process of removing unavailable calendar slots from a FlexE PHY is called "crunching" (see [OIFFLEXE1]). The following additional notes apply to Figure 2:

a. The crunched FlexE PHYs are independently transported through the transport network. The number of used (and unused) calendar slots can be different across the FlexE group. In particular, if all the calendar slots in a FlexE PHY are in use, the crunching operation leaves the original signal intact.

b. In this illustration, the different FlexE PHY(s) are transported using ODUflex containers in the transport network. These ODUflex connections can be of different rates.

c. In the most general form, G.709 Section 17.12 allows for a FlexE group consisting of m Ethernet PHY(s) to be crunched, combined, and transported using n ODUflex containers (where n can range between 1 and m). In other words, the ITU G.709 recommendation allows for (but not require the support for) the degenerate cases in which (a) each Ethernet PHY within the group is transported using its own ODUflex, and (b) all the PHY(s) are crunched, combined and transported over a single ODUflex container. If all the sub-calendar slots in a given PHY are available, it is possible to transport the content of the PHY in one of two ways: (a) as shown in Figure 2, or (b) using a FlexE unaware (i.e. PCS-codeword transparent transport) mode. The latter approach (of using FlexE unaware transport) for a few select (fully-utilized) PHYs is not attractive from the perspective of skew between the PHYs that comprise the FlexE group. For simplicity, the preferred mode of operation will be one in which the same mapping procedure is used for member PHYs of a FlexE group.
d. When the crunched FlexE PHY(s) have a rate that is identical to that of a standard Ethernet PHY, it is possible that the transport network may utilize standard ODU containers such as ODU2e, ODU4 etc. As currently defined by ITU G.709 Section 17.12, the crunched, sub-rate signal is always mapped to an ODUflex, and the mapping to a fixed rate ODU signal is not required. This option could be dropped if it results in any significant simplification.

Note: The figure may need further editing to accurately depict the signal hierarchy.
Figure 2: FlexE Aware Transport

+ Legend:
R1, R2 + Routers (supporting the FlexE clients)
NE A, Z + Transport Network Edge nodes
FlexE-psg: FlexE partial rate (sub) group signal (per G.709:17.12)
3.3. FlexE Termination - Transport

These usecases build upon the basic router-transport equipment connectivity illustrated in Figure 1. The FlexE shim layer at the router maps to the set of FlexE clients over the FlexE group, as usual. This section considers various usecases in which the equipment located at the edge of the transport network instantiates the FlexE Shim function which peers with the FlexE shim on the customer device. In the router to network direction, the transport edge node terminates the FlexE shim layer, and extracts one or more FlexE client signals, and transports them through the network. That is, these usecases are distinguished from the FlexE unaware cases in that the FlexE group, and the FlexE shim layer end at the transport network edge, and only the extracted FlexE client signals transit the optical network. In the network to router direction, the transport edge node maps a set of FlexE clients to the FlexE group (i.e. performing the same functions as the router which connects to the transport network). The various usecases differ in the combination of service endpoints in the transport network. In the FlexE termination scenarios, the distance between the FlexE Shims is limited the normal Ethernet link distance. The FlexE shims in the router, and the equipment need to support a small amount skew.

3.3.1. FlexE Client at Both endpoints

In this scenario, service consists of transporting a FlexE client through the transport network, and possibly combining this FlexE client with other FlexE clients into a FlexE group at the endpoints. The FlexE client signal can be transported in two manners within the OTN: (i) directly over one or more wavelengths (ii) mapped into an ODUflex (of the appropriate rate) and then switched across the OTN. Figure 3 illustrates the scenario involving the mapping of a FlexE client to an ODUflex envelope; this figure only shows the signal "stack" at the service endpoints, and doesn’t illustrate the switching of the ODUflex entity through the OTN. The ODUflex mapping will be beneficial in scenarios where the rate of the FlexE client is less than the capacity of a single wavelength deployed on the DWDM side of the OTN network, and allows the network operators to packet multiple FlexE client signals into the same wavelength -- thereby improving the network efficiency. Although Figure 3 illustrates the scenario in which one FlexE client is transported within the OTN, the following points should be noted:

a. When the FlexE Shim termination function recovers multiple FlexE client signals (at node A), the FlexE signals can be transported independently. In other words, it is not a requirement that all the FlexE client signals be co-routed.
b. Conversely, at the egress node, FlexE clients from different endpoints can be combined via the FlexE shim, eventually exiting the transport edge node over an Ethernet group.

---

Figure 3: FlexE termination: FlexE clients at both endpoints

### 3.3.2. Interworking of FlexE Client w/ Native Client at the other endpoint

The OIF implementation agreement [OIFFLEXE1] currently supports FlexE client signals carried over one or more 100GBASE-R PHY(s). There is a calendar of 5G timeslots associated with each PHY, and each FlexE client can make use of a number of timeslots (possibly distributed across the members of the FlexE group). This implies that the FlexE client rates are multiples of 5Gbps. When the rates of the FlexE client signals matches the MAC rates corresponding to existing Ethernet PHYs, i.e. 10GBASE-R/40GBASE-R/100GBASE-R, there is a need for the FlexE client signal to interwork with the native Ethernet client received from a single (non-FlexE capable) Ethernet PHY. This
capability is expected to be extended to any future Ethernet PHY rates that the IEEE may define in future (e.g. 25G, 50G, 200G etc.). In these cases, although the bit rate of the FlexE client matches the MAC rate of other endpoint, the 64B66B PCS codewords for the FlexE client need to be transformed (via ordered set translation) to match the specification for the specific Ethernet PHY. These details are described in Section 7.2.2 of [OIFFLEXE1] and are not elaborated any further in this document.

Figure 4 illustrates a scenario involving the interworking of a 10G FlexE client with a 10GBASE-R native Ethernet signal. In this example, the network wrapper is ODU2e.

Figure 4: FlexE client interop with Native Ethernet Client
3.3.3. Interworking of FlexE client w/ Client from OIF_MLG

As explained in the Introduction section (Section 1 OIFMLG3 [OIFMLG3] introduced support for carrying 10GE and 40GE client signals over a group of 100GBASE-R Ethernet PHY(s). While the most recent implementation agreement doesn’t call it out explicitly, it is expected that the FlexE clients (as defined in [OIFFLEXE1]), and 10GBASE-R/40GBASE-R clients supported by OIFMLG3 [OIFMLG3]) will interoperate.

Figure 5 illustrates a scenario involving the interworking of a 10G FlexE client with a 10GBASE-R client supported by an OIFMLG3 interface. In this example, the network wrapper is ODU2e.

---

Figure 5: FlexE client interop with Ethernet Client supported by MLG3
3.3.4. Back-to-Back FlexE

This section covers a degenerate FlexE termination scenario in which router1, router2, and router3 are interconnected through back-to-back FlexE groups without an intermediate transport network (see Figure 6). In this example, the FlexE SHIM at Router2 extracts one or more FlexE client signals from the FlexE group connected to Router1, and multiplexes these extracted FlexE signals into the FlexE group towards the appropriate router (e.g. Router3). Note that each of the extracted FlexE client signals can be independently routed towards its respective FlexE group.

-------------

+--------+ 2 x 100GE +---------+ 3 x 100GE +---------+
|        |           |         |           |         |
| Router1|           |         |           |         |
| FlexE  +-----------+ Router2 +-----------+ Router3 |
| Shim   |           | FlexE   +-----------+ FlexE   |
|        +-----^-----+ Shim    +-----^-----+ Shim    |
|        |     |     |         |     |     |         |
|        |     |     |         |     |     |         |
|        +--------+     +     +---------+     +     +---------+

FlexE Group           FlexE Group

-------------

Figure 6: Back-to-Back FlexE

3.3.4.1. FlexE Client BW Resizing

In the scenario presented in Figure 6, it is possible to support the FlexE client signal resizing on an end-to-end basis. Thus, for example, the resizing of the end-to-end FlexE client circuit with a scope of Router1-Router2-Router3 is accomplished by correctly coordinating the resizing operations across these two segments: Router1-Router2, Router2-Router3. The hop-by-hop FlexE client signal resizing operations across each of these segments (or hops) are accomplished by using the following FlexE overhead (as per [OIFFLEXE1]):

a. Currently active FlexE calendar (containing a list of mapping between the 5G tributary slots and the FlexE client signals

b. Future calendar to which the sender wants to transition to.
c. Calendar switch request bit (CR)
d. Calendar switch acknowledge bit (CA)

It is expected that the exact sequence of FlexE client resizing operations will be different for the cases involving bandwidth increase/decrease.

4. Requirements

This section summarizes solution requirements for the use cases described in this document to help identify the Control Plane (i.e. Routing and Signaling) extensions that may be required.

a. The solution SHALL support a FlexE group to address abovementioned use cases including FlexE unaware (where FlexE mux and demux can be separated by longer distances), FlexE aware (where FlexE mux and demux can be separated by shorter distances), and FlexE partially aware.

b. The solution SHALL support a flexible mechanism for configuring a FlexE group -- such as a signaling protocol or a SDN controller/management system with network access to the FlexE mux/demux at each end of the FlexE group.

c. The solution SHOULD support the ability to add/remove Ethernet PHYs to/from a FlexE group. In the absence of this ability, it is acceptable to permit changes to the group members only when the group has been administratively locked (and hence not providing any service).

d. The solution SHOULD allow decoupling of FlexE group’s initial configuration and bring up operation from an addition (or removal) of FlexE clients to the FlexE group. For instance, it SHOULD be possible to configure and bring up a FlexE group without any FlexE client (e.g., with all calendar slots set to unused or unavailable).

e. The solution SHALL allow adding or removing a FlexE client to a FlexE group without affecting traffic on other clients.

f. The solution SHOULD allow resizing of FlexE client BW through coordination of calendar updates within a single FlexE group. There SHOULD be no expectation that FlexE client BW resizing be hitless in all network scenarios. This capability can be supported for the Back-to-Back FlexE scenario identified in Section 3.3.4.1.
g. For the FlexE unaware case, each of the 100GBASE-R PHYs in the FlexE group SHALL be carried independently across transport network using a PCS codeword transparent mapping. All PHYs of the FlexE group SHALL be interconnected between the same two FlexE shims. The Ethernet PHYs SHOULD be carried over the same fiber route across the transport network (i.e., co-routed).

h. For the FlexE aware case, each of the 100GBASE-R PHYs in the FlexE group SHALL be carried independently across transport network. All PHYs of the FlexE group SHALL be interconnected between the same two FlexE shims. The Ethernet PHYs SHOULD be carried over the same fiber route across the transport network. In the transport network, in mux direction, the OTN mapper SHALL be able to discard unavailable slots (e.g., this can be based on static configuration as the rate of a wavelength is not expected to change in-service). In the transport network, in the demux direction, the OTN mapper SHALL be able to restore unavailable slots to match the original PHY rate.

i. For the FlexE termination case, the FlexE group SHALL be terminated at the transport network edge. It SHOULD be possible to carry (switch) each FlexE client extracted from the FlexE group independently across transport network using OTN mapping (e.g., ODUflex).

5. Framework
6. Architecture
7. Solution
8. Acknowledgements
9. IANA Considerations
   This memo includes no request to IANA.
10. Security Considerations
   None.
11. References
11.1. Normative References

11.2. Informative References


Appendix A. Additional Stuff

This becomes an Appendix.

Authors' Addresses

Iftekhar Hussain (editor)
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA

Email: IHussain@infinera.com

Radha Valiveti
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA

Email: rvaliveti@infinera.com
Khuzema Pithewan
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA
Email: kpithewan@infinera.com

Qilei Wang (editor)
ZTE
Nanjing
CN
Email: wang.qilei@zte.com.cn

Loa Andersson (editor)
Huawei
Stockholm
Sweden
Email: loa@pi.nu

Fatai Zhang
Huawei
CN
Email: zhangfatai@huawei.com

Mach Chen
Huawei
CN
Email: mach.chen@huawei.com

Jie Dong
Huawei
CN
Email: jie.dong@huawei.com
Abstract

This document specifies the GMPLS control plane requirements, framework, and architecture for the FlexE technology. The document also discusses interoperation between the GMPLS control plane for FlexE and the control plane of any networking layer using the FlexE technology as a server layer.

As different from earlier Ethernet data planes FlexE allows for decoupling of the Ethernet Physical layer (PHY) and Media Access Control layer (MAC) rates.

Study Group 15 (SG15) of the ITU-T has endorsed the FlexE Implementation Agreement from Optical Internetworking Forum (OIF) and included it, by reference, in some of their Recommendations.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 17, 2019.
Table of Contents

1. Introduction ................................................. 3
  1.1. Requirements Language ................................. 5
  1.2. Updates in the version ............................... 5
2. Terminology ................................................ 6
3. FlexE Reference Model ................................. 7
4. GMPLS Controlled FlexE ......................... 8
  4.1. Interfaces in a FlexE network ..................... 9
  4.2. Mapping of traffic in the data plane .......... 9
  4.3. The GMPLS Control Plane and the FlexE identifiers ... 10
  4.4. Operational concerns .............................. 11
  4.5. Pre-configured vs. Control Plane established LSPs in a FlexE capable network ................... 11
  4.6. Signaling Channel ..................................... 12
  4.7. MPLS LSP over the FlexE Data Plane ............... 12
  4.8. Configuring the data plane in FlexE capable nodes .......... 14
    4.8.1. Configure/Establish a FlexE Group/Link .... 14
    4.8.2. Configure/Establish a FlexE Client .......... 15
    4.8.3. Advertise FlexE Groups and FlexE Clients ... 15
5. Framework and Architecture ..................... 15
  5.1. FlexE Framework .................................... 16
  5.2. FlexE Architecture ................................ 16
    5.2.1. Architecture Components ...................... 16
    5.2.2. FlexE Layer Model .............................. 16
      5.2.2.1. FlexE Group structure .................... 17
      5.2.2.2. FlexE Client mapping ..................... 17
  6. Control Plane ....................................... 17
  6.1. GMPLS Routing ..................................... 18
  6.2. GMPLS Signaling ................................... 18
    6.2.1. LSP setup with pre-configured FlexE infrastructure ................ 19
    6.2.2. LSP setup with partially configured FlexE infrastructure ................ 20
1. Introduction

This document specifies the GMPLS control plane requirements, framework, and architecture for the FlexE technology. The FlexE control plane requirements are found in an appendix.

Prior to FlexE Ethernet MAC rates were until constrained to match the rates of the Ethernet PHY(s). FlexE, specified by the OIF, allows MAC rates to be different from PHY rates. An OIF implementation agreement [OIFFLEXE1] allows for complete decoupling of the MAC and PHY rates. This has been further extended in [OIFFLEXE2].

SG15 in ITU-T has endorsed the OIF FlexE data plane and parts of [G.872], [G.709], [G.798] and [G.8021]. The Recommendations depends on or are based on the FlexE data plane.

The FlexE implementation agreement includes support for:

a. MAC rates which are greater than the rate of a single PHY; multiple PHYs are bonded to achieve this

b. MAC rates which are less than the rate of a PHY (sub-rate)

c. support for channelization within a single PHY, or over a group of bonded PHYs.

The capabilities supported by the FlexE data plane are:

a. Support a large rate Ethernet MAC over bonded Ethernet PHYs, e.g. supporting a 200G MAC over 2 bonded 100GBASE-R PHY(s)

b. Support a sub-rate Ethernet MAC over a single Ethernet PHY, e.g. supporting a 50G MAC over a 100GBASE-R PHY
c. Support a collection of flexible Ethernet clients over a single Ethernet PHY, e.g. supporting two MACs with the rates 25Gbps, and one with rate 50G over a single 100GBASE-R PHY

d. Support a sub-rate Ethernet MAC over bonded PHYs, e.g. supporting a 150G Ethernet client over 2 bonded 100GBASE-R PHY(s)

e. Support a collection of Ethernet MAC clients over bonded Ethernet PHYs, e.g. supporting a 50G and 150G MAC over 2 bonded 100GBASE-R PHY(s)

FlexE networks feature FlexE Ethernet interfaces, for more details see Section 4.1.

From a control plane perspective, the FlexE Groups may be viewed as logical links and FlexE Clients as logical sub-interfaces (or channelized interfaces).

These logical point-to-point links may be realized in at least two different ways:

a. direct point-to-point links with no intervening transport network.

b. direct point-to-point links across a transport network transport network.

c. Ethernet PHY(s) may be transparently transported via an Optical Transport Network (OTN), as defined by ITU-T in [G.709] and [G.798].

The OTN set of client mappings has been extended to support the use cases identified in the OIF FlexE implementation agreement.

This document is a framework for the network control plane signaling and routing extensions required to establish FlexE links (FlexE Groups (PHY) and FlexE Clients (MAC)). FlexE Links may interconnect customer edge devices (CE to CE), CE to provider edge devices (PE), PE to PE, or devices at the edge to devices in the core (PE to P) or devices in the core (P to P).

Any pair of neighbouring L2 and L3 device that support FlexE interfaces may be interconnected P2P using a FlexE link (PHY and MAC). Further a device that terminates a FlexE link MUST be able to extract either the L2 or L3 payload and switch on the appropriate level, i.e. Ethernet, MPLS or IP. It should be noted that any type of switching is outside is out of scope for the FlexE specification.
FlexE CE devices may typically be L3 routers or other devices that use FlexE at layers 1 and 2 to provide point-to-point connectivity between each other.

Thus this draft considers the cases in which the two peer FlexE devices are:

- interconnecting two parts of a customer network (CE to CE).
- at the edge of the customer network (CE) and the close edge of the provider network (PE to CE).
- opposite edges of the FlexE capable network (PE to PE).
- at the edge of the FlexE network PE interconnected to a provider device (PE to P).
- interconnecting two provider devices (P to P).

This list of deployment cases will help identify the GMPLS control plane (i.e. routing and signaling) extensions that may be required to support establishment of FlexE services.

1.1. Requirements Language

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

1.2. Updates in the version

This section will be removed before posting.

1. Following a suggestion from Daniele the FlexE Control Plane Requirements has been moved to an appendix.

2. There are still some of the comments from Daniele that might need to be addressed, but we have had a pretty large overlap in comments, so the intention is that all should be addressed.

3. The terms Ethernet Interface and Ethernet sub-Interface has been re-introduced in relation to FlexE Group and FlexE Client respectively.
4. Except for some spelling corrections Section 5 to Section 7 are virtually untouched, though it is likely that some of the changes in the earlier parts of the document will have to be reflected into those sections also.

2. Terminology

a. CE (Customer Edge): the group of functions that support the termination/origination of data received from or sent to the network. Sometimes the term CE device is used.

b. controller: a joint term for any entity that may set up a LSP, FlexE Group or FlexE Client, e.g. a control plane, centralized controller, YANG model or management system.

c. crunch: the term crunch in the context of OTN networks and FlexE links is used when e.g. unavailable calendar slots are not transported across the OTN network, but are removed at the ingress and recreated at the egress.

d. Ethernet PHY: an entity representing Physical Coding Sublayer (PCS), Physical Media Attachment (PMA), and Physical Media Dependent (PMD) layers.

e. FlexE Calendar: The total capacity of a FlexE Group is represented as a collection of slots which have a granularity of 5Gbps. The calendar for a FlexE Group composed of n 100G PHYs is represented as an array of 20n slots (each representing 5Gbps of bandwidth). This calendar is partitioned into sub-calendars, with 20 slots per 100G PHY. Each FlexE Client is mapped into one or more calendar slots (based on the bandwidth the FlexE Client flow will need).

f. FlexE Channelized sub-Interface, the channelized Ethernet sub-interface realized by the FlexE Client.

g. FlexE Client: An Ethernet flow based on a MAC data rate that may or may not correspond to any Ethernet PHY rate.

h. FlexE Group: A FlexE Group is composed of from 1 to n Ethernet PHYs. In the first version of FlexE each PHY is identified by a number in the range {1-254}.

i. FlexE Interface, the Ethernet interface realized the FlexE Group

j. FlexE Shim: the layer that maps or demaps the FlexE Client flows carried over a FlexE Group.
k. LMP: Link Management Protocol
l. LSP: Label Switched Path
m. OIF: Optical Internetworking Forum
n. OTN: Optical Transport Network
o. PE: Provider Edge (device) the term is used for the functions needed at the edge of a provider network or the device to which these functions are allocated.
p. P: Provider (device), the term is used for the functions needed in the core of a provider network or the device to which these functions are allocated.
r. TE: Traffic Engineering
s. TED: Traffic Engineering Database

3. FlexE Reference Model

The figure below gives a simplified FlexE reference model.
The services offered by Flexible Ethernet are essentially the same as for traditional Ethernet, connection less Ethernet transport. In essence the FlexE interfaces and links may be viewed as any other Ethernet interfaces or links. However, it is possible to capture additional TE information in the Traffic Engineering Data Base showing unique characteristics of FlexE channelized interfaces and links. This makes it possible for the control plane to strategically use FlexE networks to support advanced TE.

4. GMPLS Controlled FlexE

The high level goals for using a GMPLS control plane for FlexE can be summarized as:

- Set up a FlexE Group
- Set up a FlexE Client
- Advertise the TE information of FlexE Groups and FlexE Clients
Set up of a higher layer LSPs that require to be (or would have significant benefits to) be run over a FlexE infrastructure.

Decoupling PHY and MAC bandwidth opens up some interesting features for networks that features FlexE links. By establishing several FlexE Clients with bandwidth that are part of the bandwidth of the FlexE Group, it is possible to create channels between to nodes.

By controlling the mapping a user packets (or frames) to these channels it is possible to create bandwidth that are dedicated for special purposes, and that can’t be infringed on by packets (or frames) that does not satisfy this mapping.

4.1. Interfaces in a FlexE network

FlexE Ethernet interfaces are realized by the means of a basic building block. The same building block is used for a single PHY and when the PHY’s are bonded. The building block consists of two FlexE Shim functions (see Section 5.2.2.2) and a logical point to point link. The FlexE Shim functions are located at each end of the logical point to point link. This link carries the Ethernet PHY signals between the two FlexE Shim Functions.

4.2. Mapping of traffic in the data plane

An example of which data plane mappings takes palace when an upper layer, e.g. IP or MPLS, send packets over a FlexE interfaces is shown in Figure 2.
Figure 2: Traffic Mapping

In the mapping steps indicated in Figure 2 only one step in the mapping is visible by each layer.

- The MPLS layer knows from the IP address, which MPLS label stack to encapsulate the IP packet in.
- The MPLS layer also knows which MPLS label(s) that maps to which FlexE Client.
- The FlexE layer also knows from the FlexE Client Identifier, which calendar slots the packet will be transferred over.
- The FlexE layer knows which FlexE Group a certain set of calendar slots belongs to.

4.3. The GMPLS Control Plane and the FlexE identifiers

This section lists some of the procedures and actions on FlexE Interface Identifiers that a GMPLS Control plane need to perform. Also, a centralized controller, YANG model or a management system that are used to establish interfaces and links need to perform the same actions.

The FlexE Group Identifier and the FlexE Client Identifier, included in the overhead of each frame sent over a FlexE Interface or sub-Interface, indicates a particular Group or Client.

When the Control Plane, a centralized controller, a YANG model or a management system sets up a FlexE Interface at least the bandwidth...
has to be included in the setup message. The FlexE system returns the FlexE Group Identifier in the response message.

When a channelized sub-interface is set up, the party that initiates the setup includes the Interface (FlexE Group) Identifier over which the sub-Interface will be established, and the bandwidth requested for the sub-interface. The FlexE system returns the FlexE Client Identifier.

The identifiers received by the party that initiate a setup of an FlexE Interfaces are used, by a controller, to set up FlexE sub-interfaces.

The identifiers received by the party that initiate a setup of an FlexE sub-Interfaces are used, e.g. to map an MPLS label to the correct FlexE sub-interfaces.

4.4. Operational concerns

When operating a links in a FlexE network it is likely that an operator would like to split the FlexE Interface in sub-Interfaces used for best effort traffic and sub-Interfaces for dedicated for special purposes. An example would be when there is a 100 Gbit/s FlexE are split in to five 10 Gbit/s sub-interfaces and one 50 Gbit/s sub-interface. The 50 Gbit/s sub-interface could be used best effort traffic, the five 10 Gbit/s could be used for dedicated traffic.

In such cases it is conceivable that packets/frames that have a matching key will be put on a specific sub-Interface, while traffic that do not have a matching key will be put on the best effort sub-interface.

4.5. Pre-configured vs. Control Plane established LSPs in a FlexE capable network

The FlexE infrastructure may be established in three different ways

- The FlexE Groups and FlexE Client may be pre-configured
- Only the FlexE Groups may be pre-configured, while the setup of the FlexE Client is triggered by the request to setup a MPLS LSP.
- The setup of both FlexE Group and FlexE Client may be triggered by the request to setup an MPLS LSP.

In the case the FlexE Groups and FlexE Clients are preconfigured the FlexE capable nodes need to have the ability to announce the preconfigured FlexE Client and/or FlexE Groups as if they were LSPs.
4.6. Signaling Channel

In the type of equipment for which FlexE was first specified an out of band signaling channel is not commonly available. If that is the case, and the GMPLS FlexE control plane will be used, the FlexE Group will have to setup by e.g. a management system and a FlexE Client on that FlexE Group (also configured) will have to allocated as a signaling channel.

Further details of the setup of the FlexE Groups, FlexE Clients and MPLS LSPs over a FlexE infrastructure will be found in Section 6.2.

4.7. MPLS LSP over the FlexE Data Plane

FlexE is a true link layer technology, i.e. it is not switched, this means that the FlexE Groups and FlexE Clients are terminated on the next-hop node, and that the switching needs to take place on a higher layer.

The FlexE technology can be used to establish link layer connectivity with high and deterministic bandwidth. However, there is no way described in the FlexE specification to, in a deterministic way, allocate certain traffic to a specific FlexE Client. Control of the FlexE link layer by a GMPLS control plane can achieve this.

A GMPLS controlled FlexE capable node may be thought of using the traditional model of a node with a separation between control and data plane.
The GMPLS control plane will speak extended standard GMPLS protocols with its neighbours and peers.
Figure 4 describes how an MPLS LSP is mapped over a FlexE Client and FlexE Group.

4.8. Configuring the data plane in FlexE capable nodes

In Figure 4 we show an LSP, a FlexE Client and a FlexE Group, the LSP is there because while the FlexE Channel and Group are not switched, switching in our example takes place on the LSP level. This section will discuss establishment of FlexE Clients and Groups, and mapping of the LSP onto a FlexE Client.

The establishment of a LSP over a FlexE system is very similar to how this is done in any other system. Building on information gathered through the routing system and using the GMPLS signaling to establish the LSP.

4.8.1. Configure/Establish a FlexE Group/Link

Consider the setup of a FlexE Group between node A and B, corresponding to the row of U's from node A to B in Figure 4. The FlexE Group is considered to consist of n PHYs, but does not have any FlexE Clients defined from start.

When this is done by the GMPLS control plane, two conditions need to be fulfilled (1) there need to be a data channel defined between node A and B; and (2) a FlexE capable IGP-TE protocol needs to be running in the network.

Node A will send an RSVP-TE message to node B with the information describing the FlexE Group to be setup. This information might be thought of as the "FlexE Group Label" (or part of the FlexE label). It will contain at least the following information:

- A FlexE Group Identifier (FGid).
- The number of active FlexE Channels (numFC), where 0 indicates that zero clients are active.
- Number of PHYs that the FlexE Group is composed of, for each PHY
  * PHY identifier
  * PHY bandwidth
  * slot granularity/number of slots
  * available and unavailable slots
When node B receives the RSVP-TE message it checks that it can setup the requested FlexE Group. If the check turns positive, node send an acknowledgment to node A and the FlexE Group is setup.

A more detailed description of how to setup a FlexE Group, will be included in the draft dealing with signaling in detail.

4.8.2. Configure/Establish a FlexE Client

Consider the situation where a FlexE Group is already established (as described in Section 4.8.1) and an m G FlexE Client is needed. Similar to the establishment of the FlexE Group, node A will send a RSVP-TE message to node B.

This RSVP-TE message include at least the following information:

- FlexE Group Identifier
- FlexE Client Identifier
- from which PHYs the slots will allocated, i.e. slots might come from more than one PHY.
- Information per PHY
  - PHY bandwidth
  - slot granularity
  - available/unavailable slots
  - allocated slots

A more detailed description of how to setup a FlexE Channel, will be included in the draft dealing with signaling in detail.

4.8.3. Advertise FlexE Groups and FlexE Clients

Once the FlexE Group and FlexE Clients are configured they can be advertised into the routing system as normal routing adjacencies, including the FlexE specific TE information.

5. Framework and Architecture

This section discusses FlexE framework and architecture. Framework is taken to mean how FlexE interoperates with other parts of the data communication system. Architecture is taken to mean how functional groups and elements within FlexE work together to deliver the
expected FlexE services. Framework is taken to mean how FlexE interacts with its environment.

5.1. FlexE Framework

The service offered by Flexible Ethernet is a transport service very similar (or even identical) to the service offered by Ethernet.

There are two major additions supported by FlexE:

- FlexE is intended to support high bandwidth and FlexE can offer granular bandwidth from 5Gbits/s and a bandwidth as high as the FlexE Group allows.
- As FlexE Groups and clients are set up as a configuration activity, by a centralized controller or by a GMPLS control plane the service is connection oriented.

5.2. FlexE Architecture

5.2.1. Architecture Components

This section discusses the different parts of FlexE signaling and routing and how these parts interoperate.

The FlexE routing mechanism is used to provide resource available information for setup of higher layer LSPs, like Ethernet PHYs’ information, partial-rate support information. Based on the resource available information advertised by routing protocol, an end-to-end FlexE connection is computed, and then the signaling protocol is used to set up the end-to-end connection.

FlexE signaling mechanism is used to setup LSPs.

MPLS forwarding over a FlexE infrastructure is different from forwarding over other infrastructures. When MPLS runs over a FlexE infrastructure it is possible that there are more than FlexE Client that meet the next-hop requirements, often it is possible to use any suitable FlexE Client for a hop between two nodes. If the mapping between a MPLS encapsulated packet and the FlexE Client, this mapping need to be explicit when the LSP is set up, and the MPLS label will be used to find the correct FlexE Client.

5.2.2. FlexE Layer Model

The FlexE layer model is similar Ethernet model, the Ethernet PHY layer corresponds to the "FlexE Group", and the MAC layer corresponds to the "FlexE Client".
As different from earlier Ethernet the combination of Flexe Group and Client allows for a huge freedom when it comes to define the bandwidth of an Ethernet connectivity.

5.2.2.1. FlexE Group structure

The FlexE Group might be supported by virtually any transport network, including the Ethernet PHY. While the Ethernet PHY offers a fixed bandwidth the FlexE Group has been structured into 5 Gbit/s slots. This means that the FlexE Group can support FlexE Clients of a variety of bandwidths.

The first version is defined for 20 slots of 5 Gbit/s over a 100 Gbit/s PHY. The 100 Gbit/s PHYs can be bonded to give higher bandwidth.

5.2.2.2. FlexE Client mapping

A FlexE Client is an Ethernet flow based on a MAC data rate that may or may not correspond to any Ethernet PHY rate. The FlexE Shim is the layer that maps or demaps the FlexE Client flows carried over a FlexE Group. As defined in [OIFFLEXE1], MAC rates of 10, 40, and any multiple of 25 Gbit/s are supported. This means that if there is a 100 Gbit/s FlexE Group between A and B, a FlexE Client of 10, 25, 40, 50, 75 and 100 Gbit/s can be created.

However, by bonding, for example 5 PHYs of 100 Gbit/s to a single FlexE Group, FlexE Clients of 500 Gbit/s can be supported.

6. Control Plane

This section discusses the procedures and extensions needed to the GMPLS Control Plane to establish FlexE LSPs.

There are several ways to establish FlexE Groups, allocate slots for FlexE Clients, and setup higher layer LSPs. A configuration tool, a centralized controller or the GMPLS control plane can all be used.

To create the FlexE GMPLS control plane Groups, FlexE Clients and higher layer LSPs, extensions to the following protocols may be needed:

- "RSVP-TE: Extensions to RSVP for LSP Tunnels" (RSVP-TE) [RFC3209]
- "Link Management Protocol" (LMP) [RFC4204]
- "Path Computation Element (PCE) Communication Protocol" (PCEP) [RFC5440]
A FlexE control plane YANG model will also be needed.

Section 6.2 and Section 6.1 discuss the role of the GMPLS control plane when primarily setting up LSPs.

When discussing the signaling and routing procedures we assume that the FlexE Group has been established prior to executing the procedures needed to establish an LSP. Technically it is possible to establish FlexE Group, allocate FlexE Client slots and LSP with a single exchange of GMPLS signaling messages.

6.1. GMPLS Routing

To establish an LSP the Traffic Engineering (TE) information is the most critical information, e.g. resource utilization on interfaces and link, including the availability of slots on the FlexE Groups. The GMPLS routing protocols need to be extended to handle this information. The Traffic Engineering Database (TED) will keep an updated version of this information.

The FlexE capable nodes will be identified by IP-addresses, and the routing and traffic engineering information will be flooded to all nodes within the routing domain using TCP/IP.

When an LSP over the FlexE infrastructure is about to be setup, e.g. R1 - R4 - R5 in Figure 5 the information in the TED is used verify that resources are available. When it is conformed that the LSP is established the TED is updated, marking the resources used for the new LSP as used. Similarly, when a LSP is taken down the resources are marked as free.

6.2. GMPLS Signaling

As described in Section 4 the state of the FlexE infrastructure may effect the actions needed to setup an LSP in a FlexE capable network. The FlexE infrastructure maybe be:

1. fully pre-configured
2. partially pre-configured, i.e. the FlexE Group may be pre-configured, but not the FlexE Clients

3. not pre-configured, i.e. the setup of FlexE Group and FlexE Client will be triggered because of the request to setup an LSP.

Figure 5 will be used to illustrate the different cases.

```
+----+
| R1 +---------------------+
+----+

+----+           +----+               +----+
| R2 +------------------+ R4 +-------------------------+ R5 |
+----+           +----+               +----+

+----+           +----+
| R3 +---------------------+ PHY R1 to R4 100 Gbit(s)
+----+ PHY R2 to R4 100 Gbit(s)
| PHY R3 to R4 100 Gbit(s)
+ PHY R4 to R5 200 Gbit(s)
```

Figure 5: FlexE LSP Example

The text in Section 6.2 is not a specification of the GMPLS signaling extensions for FlexE capable network, it is a description to illustrate the expected features of such a protocol. Nor do we discuss failure scenarios.

6.2.1. LSP setup with pre-configured FlexE infrastructure

In this first example, referencing Figure 5, one 100 Gbit/s FlexE Group is configured between R1 and R4, between R2 and R4, and between R3 and R4. Between R4 and R5 there is a 200 Gbit/s FlexE Group.

Over each 100 Gbit/s FlexE Group there are four 5 Gbit/s, two 20 Gbit/s and one 40 Gbit/s FlexE Clients configured. Over the 200 Gbit/s FlexE Group there are eight 5 Gbit/s, four 20 Gbit/s and two 40 Gbit/s FlexE Clients configured.

One of the 5 Gbit/s FlexE Clients on each FlexE Groups are used as signaling channel.
To establish the for example a 200 Mbit/s MPLS LSP the normal GMPLS request/response procedures are followed. R1 sends the request to R4, R4 allocate resources on one of the FlexE Clients, forward the request to R5. R5 responds to R4 indicating the label and the FlexE Client the traffic should be sent over, R4 does the same for R1.

The only difference between the standard signaling and what happens here is that there the assigned label will be used to find the right FlexE Client.

6.2.2. LSP setup with partially configured FlexE infrastructure

In the second example, also referencing Figure 5, the FlexE Groups are setup in the same way as in the first example, however only one 5 Gbit/s FlexE Client per FlexE Group are established by configuration. This FlexE Client will be used for signaling.

When preparing to send the request that a 5 Gbit/s MPLS LSP shall be set up R1 discovers that there are no feasible FlexE Client between R1 and R4. R1 therefore sends the request to establish such a FlexE Client, when receiving the request R4 allocates resources for the FlexE Client on the FlexE Group. There may be different strategies for allocating the bandwidth for this FlexE Client. Such strategies are out of scope for this document. R1 then sends the information about the FlexE Client to R1, and both ends establish the FlexE Client.

When the FlexE Client between R1 and R4 is established, R1 proceeds to send the request for an MPLS LSP to R4. R4 will discover that a feasible FlexE Client is missing between R4 and R5. The same procedure s for setting up the FlexE Client between R1 and R4 is repeated for R4 and R5. When there is a feasible FlexE Client available the signaling to set up the MPLS LSP continues as normal.

The label allocated for the MPLS LSP will be used to find the correct FlexE Client.

When a FlexE Clients is set up in this way they can be announced into the routing system in two different ways. First, they can be made generally available, i.e. it will be free to use for anyone that want to set up LSPs over the FlexE Group between R1 and R4 and between R4 and R5. Second, the use of the FlexE Clients may be restricted to the application that initially did set up the FlexE Client.
6.2.3. LSP setup with non-configured FlexE infrastructure

This example also refers to Figure 5 as different from the earlier example no FlexE Group or FlexE Client configuration is done prior to the first request for an MPLS LSP over the FlexE infrastructure.

To make the set up of LSPs in a FlexE network where no FlexE Groups or FlexE Clients have been configured two conditions need to be fulfilled. First an out of band signaling channel must be available. Second the FlexE Capabilities must be announced in to the IGP and/or centralized controller.

If these two conditions are fulfilled, the set up of an MPLS LSP progress pretty much as in the partially configured network. The difference is that the set up of both the FlexE Group and FlexE Client are triggered by the request to set up an MPLS LSP.

As in the partially configured case FlexE Clients can be announced into the routing system in two different modes, either they are generally available. It or they are reserved for the applications that first established them.

6.2.4. Packet Label Switching Data Plane

This section discusses how the FlexE LSP data plane works. In general it can be said that the interface offered by the FlexE Shim and the FlexE Client is equivalent to the interface offered by the Ethernet MAC.

Figure 6 below illustrates the FlexE packet switching data plane procedures.
The data plane processes packets like this:

- The LSP encapsulating and forwarding function in node R1 receives a packet that needs to be encapsulated as an MPLS packet with the label "a". The label "a" is used to figure out which FlexE emulated Ethernet interfaces the label encapsulated packet need to be forwarded over.

- The Ethernet interfaces, by means of FlexE transport, forwards the packet to node R3. Node R3 swaps the label "a" to label "b" and uses "b" to decide over which interface to send the packet.

- Node R3 forwards the packet to node R, which terminates the LSP.

Sending MPLS encapsulated packets over a FlexE Client is similar to send them over an Ethernet 802.1 interface. The critical differences are:

- FlexE channelized sub-interfaces guarantee a deterministic bandwidth for an LSP.

- When a application that originally establish a FlexE Client reserve it for use by that application only, it is possible to create uninfringeable bandwidth end-to-end for an MPLS LSP.

- FlexE infrastructure allows for creating very large end to end bandwidth
7. Operations, Administration, and Maintenance (OAM)
   To be added in a later version.

8. Acknowledgements

9. IANA Considerations
   This memo includes no request to IANA.
   Note to the RFC Editor: This section should be removed before publishing.

10. Security Considerations
    To be added in a later version.

11. Contributors
    Khuzema Pithewan, Infinera Corp, kpithewan@infinera.com
    Fatai Zhang, Huawei, zhangfatai@huawei.com
    Jie Dong, Huawei, jie.dong@huawei.com
    Zongpeng Du, Huawei, duzongpeng@huawei.com
    Xian Zhang, Huawei, zhang.xian@huawei.com
    James Huang, Huawei, james.huang@huawei.com
    Qiwen Zhong, Huawei, zhongqiwen@huawei.com
    Yongqing Zhu China Telecom zhuyq@gsta.com
    Huanan Chen China Telecom chenhuanan@gsta.com

12. References

12.1. Normative References

   [G.709]   ITU, "Optical Transport Network Interfaces
            (http://www.itu.int/rec/T-REC-G.709-201606-P/en)", July
            2016.


[OIFFLEXE1] OIF, "FLex Ethernet Implementation Agreement Version 1.0 (OIF-FLEXE-01.0)", March 2016.


12.2. Informative References


Appendix A. Requirements

This section summarizes the signaling and routing requirements for a FlexE control plane, with respect to establishing FlexE Groups, FlexE Clients and MPLS LSPs that require support from an FlexE infrastructure.

Req-1  The FlexE control plane SHALL support the creation of FlexE Groups.

* A FlexE Groups consist one or more 100GE Ethernet PHY(s).
  In the first version of FlexE the number of PHYs are in the range of 1 to 254.

* This requirement can be met by several methods, e.g. routing and signaling protocols, a centralized controller or a management system.

    Any such method need to have network access to the FlexE shims at each of the Ethernet PHY(s) termination points.

Req-2  The FlexE control plane SHALL have the ability to delete a FlexE Group.

Req-3  The FlexE control plane SHALL have the ability to initiate an administratively lock or unlock of a FlexE Group.

* This ability is needed e.g. for executing the next requirement.

Req-4  When a FlexE Group has been administratively looked is SHALL be possible to add PHYs to an operational FlexE Group.

Req-5  When a FlexE Group has been administratively looked is SHALL be possible to remove PHYs from an operational FlexE Group.
Req-6  The FlexE control plane SHALL support the ability to collect, advertise and discover information about FlexE capable nodes, including the TE information the FlexE Groups and FlexE Clients the nodes support.

Note: In essence correct, but something is backward. Need to think.

Req-7  The FlexE control plane SHALL allow the addition (or removal) of one or more FlexE clients to a FlexE Group. The addition (or removal) of a FlexE Client flow SHALL NOT affect the services of the other FlexE Client signals.

Req-8  The FlexE control plane SHALL, though this MAY not be possible in all network scenarios, support FlexE Client flow resizing without affecting any existing FlexE Clients within the same FlexE Group.

Req-9  The FlexE control plane SHALL support establishment of MPLS LSPs that requires the support of a FlexE infrastructure.

Authors’ Addresses

Iftekhar Hussain
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA

Email: IHussain@infinera.com

Radha Valiveti
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA

Email: rvaliveti@infinera.com

Qilei Wang
ZTE
Nanjing
CN

Email: wang.qilei@zte.com.cn
YANG Data Model for Layer 3 TE Topologies
draft-liu-teas-yang-l3-te-topo-05

Abstract

This document defines a YANG data model for layer 3 traffic engineering topologies.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at 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 3, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of
1. Introduction

This document defines a YANG [RFC7950] data model for describing the relationship between a layer 3 network topology [I-D.ietf-i2rs-yang-l3-topology] and a TE topology [I-D.ietf-teas-yang-te-topo].

When traffic engineering is enabled on a layer 3 network topology, there will be a corresponding TE topology. The TE topology may or may not be congruent to the layer 3 network topology. When such a congruent TE topology exists, there will be a one-to-one association between the one modeling element in the layer 3 topology to another element in the TE topology. When such a congruent TE topology does not exist, the association will not be one-to-one. This YANG data model allows both cases.
1.1. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119].

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

- augment
- data model
- data node

2. Modeling Considerations

2.1. Relationship Between Layer 3 Topology and TE topology

In general, layer 3 network topology model and TE topology model can be used independently. When traffic engineering is enabled on a layer 3 network topology, there will be associations between objects in layer 3 network topologies and objects in TE topologies. The properties of these relations are:

- The associations are between objects of the same class, i.e. node to node or link to link.
- The multiplicity of such an association is: 0..1 to 0..1. An object in a layer 3 network may have zero or one associated object in the corresponding TE network.

2.2. Relationship Modeling

YANG data type leafref is used to model the association relationship between a layer 3 network topology and a TE topology. YANG must statements are used to enforce the referenced objects are in the topologies of proper type.

3. Model Structure

3.1. Layer 3 TE Topology Module

The model tree structure of the layer 3 TE topology module is as shown below:
module: ietf-13-te-topology
  augment /nd:networks/nd:network/nd:network-types
   +--rw l3-te!
  augment /nd:networks/nd:network/l3t:l3-topology-attributes:
   +--rw node-ref? -> /nw:networks
  augment /nd:networks/nd:network/nd:node/l3t:l3-node-attributes:
   +--rw node-ref? -> /nw:networks
  augment /nd:networks/nd:network/nd:node/lnk:termination-point
   /l3t:l3-termination-point-attributes:
    +--rw tp-ref?     -> /nw:networks/network[nw:network-id=current()]/node/node-id
   +--rw network-ref? -> /nw:networks
  augment /nd:networks/nd:network/lnk:link/l3t:l3-link-attributes:
   +--rw link-ref?   -> /nw:networks/network[nw:network-id=current()]/node/node-id
   +--rw network-ref? -> /nw:networks

3.2. Packet Switching TE Topology Module

This is an augmentation to base TE topology model.

module: ietf-te-topology-packet
  augment /nd:networks/nd:network/nd:node/tet:te
   /tet:te-node-attributes/tet:connectivity-matrices:
    +--rw measurement
     | +--rw unidirectional-delay?      uint32
     | +--rw unidirectional-min-delay?  uint32
     | +--rw unidirectional-max-delay?  uint32
     | +--rw unidirectional-delay-variation?  uint32
     | +--rw unidirectional-packet-loss? decimal64
     | +--rw unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float32
     | +--rw unidirectional-available-bandwidth? rt-types:bandwidth-ieee-float32
     | +--rw unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float32
---rw normality
  ---rw unidirectional-delay?
types:performance-metric-normality
  ---rw unidirectional-min-delay?
types:performance-metric-normality
  ---rw unidirectional-max-delay?
types:performance-metric-normality
  ---rw unidirectional-delay-variation?
types:performance-metric-normality
  ---rw unidirectional-packet-loss?
types:performance-metric-normality
  +---rw unidirectional-residual-bandwidth?
types:performance-metric-normality
  +---rw unidirectional-available-bandwidth?
types:performance-metric-normality
  +---rw unidirectional-utilized-bandwidth?
types:performance-metric-normality
  ---rw throttle
    ---rw unidirectional-delay-offset?  uint32
    ---rw measure-interval?            uint32
    ---rw advertisement-interval?      uint32
    ---rw suppression-interval?        uint32
    ---rw threshold-out
      ---rw unidirectional-delay?       uint32
      ---rw unidirectional-min-delay?   uint32
      ---rw unidirectional-max-delay?   uint32
      ---rw unidirectional-delay-variation?  uint32
      ---rw unidirectional-packet-loss? decimal64
      +---rw unidirectional-residual-bandwidth?
types:bandwidth-ieee-float32
    ---rw unidirectional-available-bandwidth?
types:bandwidth-ieee-float32
    +---rw unidirectional-utilized-bandwidth?
types:bandwidth-ieee-float32
    ---rw threshold-in
      ---rw unidirectional-delay?       uint32
      ---rw unidirectional-min-delay?   uint32
      ---rw unidirectional-max-delay?   uint32
      ---rw unidirectional-delay-variation?  uint32
      ---rw unidirectional-packet-loss? decimal64
      +---rw unidirectional-residual-bandwidth?
types:bandwidth-ieee-float32
    ---rw unidirectional-available-bandwidth?
types:bandwidth-ieee-float32
    +---rw unidirectional-utilized-bandwidth?
types:bandwidth-ieee-float32
    ---rw threshold-accelerated-advertisement
    +---rw unidirectional-delay?       uint32
---rw unidirectional-min-delay?     uint32
---rw unidirectional-max-delay?     uint32
---rw unidirectional-delay-variation?  uint32
---rw unidirectional-packet-loss?    decimal64
---rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
---rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
---rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
augment /nd:networks/nd:network/nd:node/tet:te
/tet:node-attributes/tet:connectivity-matrices
/tet:connectivity-matrix:
  ---rw performance-metric
    ---rw measurement
      | ---rw unidirectional-delay?    uint32
      | ---rw unidirectional-min-delay? uint32
      | ---rw unidirectional-max-delay? uint32
      | ---rw unidirectional-delay-variation?  uint32
      | ---rw unidirectional-packet-loss?    decimal64
      | ---rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
    | ---rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
    | ---rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
  ---rw normality
    | ---rw unidirectional-delay?
te-types:performance-metric-normality
    | ---rw unidirectional-min-delay?
te-types:performance-metric-normality
    | ---rw unidirectional-max-delay?
te-types:performance-metric-normality
    | ---rw unidirectional-delay-variation?
te-types:performance-metric-normality
    | ---rw unidirectional-packet-loss?
te-types:performance-metric-normality
    | ---rw unidirectional-residual-bandwidth?
te-types:performance-metric-normality
    | ---rw unidirectional-available-bandwidth?
te-types:performance-metric-normality
    | ---rw unidirectional-utilized-bandwidth?
te-types:performance-metric-normality
  ---rw throttle
    | ---rw unidirectional-delay-offset?  uint32
    | ---rw measure-interval?            uint32
    | ---rw advertisement-interval?      uint32
    | ---rw suppression-interval?        uint32
+--rw threshold-out
 |  +--rw unidirectional-delay?  uint32
 |  +--rw unidirectional-min-delay?  uint32
 |  +--rw unidirectional-max-delay?  uint32
 |  +--rw unidirectional-delay-variation?  uint32
 |  +--rw unidirectional-packet-loss?  decimal64
 |  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32

+--rw threshold-in
 |  +--rw unidirectional-delay?  uint32
 |  +--rw unidirectional-min-delay?  uint32
 |  +--rw unidirectional-max-delay?  uint32
 |  +--rw unidirectional-delay-variation?  uint32
 |  +--rw unidirectional-packet-loss?  decimal64
 |  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32

+--rw threshold-accelerated-advertisement
 |  +--rw unidirectional-delay?  uint32
 |  +--rw unidirectional-min-delay?  uint32
 |  +--rw unidirectional-max-delay?  uint32
 |  +--rw unidirectional-delay-variation?  uint32
 |  +--rw unidirectional-packet-loss?  decimal64
 |  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32

  +--ro performance-metric
   |  +--ro measurement
    |   |  +--ro unidirectional-delay?  uint32
    |   |  +--ro unidirectional-min-delay?  uint32
    |   |  +--ro unidirectional-max-delay?  uint32
    |   |  +--ro unidirectional-delay-variation?  uint32
    |   |  +--ro unidirectional-packet-loss?  decimal64
    |   |  +--ro unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
   |  +--ro unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
  +--ro normality
    |  +--ro unidirectional-delay?
te-types:performance-metric-normality
    |  +--ro unidirectional-min-delay?
te-types:performance-metric-normality
    |  +--ro unidirectional-max-delay?
te-types:performance-metric-normality
    |  +--ro unidirectional-delay-variation?
te-types:performance-metric-normality
    |  +--ro unidirectional-packet-loss?
te-types:performance-metric-normality
    |  +--ro unidirectional-residual-bandwidth?
te-types:performance-metric-normality
    |  +--ro unidirectional-available-bandwidth?
te-types:performance-metric-normality
    |  +--ro unidirectional-utilized-bandwidth?
te-types:performance-metric-normality
    +--ro throttle
      +--ro unidirectional-delay-offset?  uint32
      +--ro measure-interval?  uint32
      +--ro advertisement-interval?  uint32
      +--ro suppression-interval?  uint32
      +--ro threshold-out
        |  +--ro unidirectional-delay?  uint32
        |  +--ro unidirectional-min-delay?  uint32
        |  +--ro unidirectional-max-delay?  uint32
        |  +--ro unidirectional-delay-variation?  uint32
        |  +--ro unidirectional-packet-loss?  decimal64
        +--ro unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
  +--ro unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
  +--ro threshold-in
    |  +--ro unidirectional-delay?  uint32
    |  +--ro unidirectional-min-delay?  uint32
    |  +--ro unidirectional-max-delay?  uint32
    |  +--ro unidirectional-delay-variation?  uint32
    |  +--ro unidirectional-packet-loss?  decimal64
    +--ro unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
  +--ro threshold-accelerated-advertisement
    +--ro unidirectional-delay?          uint32
    +--ro unidirectional-min-delay?      uint32
    +--ro unidirectional-max-delay?      uint32
    +--ro unidirectional-delay-variation? uint32
    +--ro unidirectional-packet-loss?    decimal64
    +--ro unidirectional-residual-bandwidth?
  rt-types:bandwidth-ieee-float32
    +--ro unidirectional-available-bandwidth?
  rt-types:bandwidth-ieee-float32
    +--ro unidirectional-utilized-bandwidth?
  rt-types:bandwidth-ieee-float32
    augment /nd:nets/nd:network/nd:node/tet:te
          /tet:information-source-entry/tet:connectivity-matrices
          /tet:connectivity-matrix:
            +--ro performance-metric
              +--ro measurement
                +--ro unidirectional-delay?          uint32
                +--ro unidirectional-min-delay?      uint32
                +--ro unidirectional-max-delay?      uint32
                +--ro unidirectional-delay-variation? uint32
                +--ro unidirectional-packet-loss?    decimal64
                +--ro unidirectional-residual-bandwidth?
              rt-types:performance-metric-normality
                +--ro unidirectional-delay?
                te-types:performance-metric-normality
                +--ro unidirectional-min-delay?
                te-types:performance-metric-normality
                +--ro unidirectional-max-delay?
                te-types:performance-metric-normality
                +--ro unidirectional-delay-variation?
                te-types:performance-metric-normality
                +--ro unidirectional-packet-loss?
                te-types:performance-metric-normality
                +--ro unidirectional-residual-bandwidth?
                te-types:performance-metric-normality
                +--ro unidirectional-available-bandwidth?
                te-types:performance-metric-normality
                +--ro unidirectional-utilized-bandwidth?
  te-types:performance-metric-normality
    +--ro throttle
      +--ro unidirectional-delay-offset?    uint32
+--ro measure-interval?                      uint32
+--ro advertisement-interval?                uint32
+--ro suppression-interval?                  uint32
+--ro threshold-out
  |  +--ro unidirectional-delay?               uint32
  |  +--ro unidirectional-min-delay?           uint32
  |  +--ro unidirectional-max-delay?           uint32
  |  +--ro unidirectional-delay-variation?      uint32
  |  +--ro unidirectional-packet-loss?         decimal64
  |  +--ro unidirectional-residual-bandwidth?
  rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-available-bandwidth?
  rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-utilized-bandwidth?
  rt-types:bandwidth-ieee-float32
+--ro threshold-in
  |  +--ro unidirectional-delay?               uint32
  |  +--ro unidirectional-min-delay?           uint32
  |  +--ro unidirectional-max-delay?           uint32
  |  +--ro unidirectional-delay-variation?      uint32
  |  +--ro unidirectional-packet-loss?         decimal64
  |  +--ro unidirectional-residual-bandwidth?
  rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-available-bandwidth?
  rt-types:bandwidth-ieee-float32
  |  +--ro unidirectional-utilized-bandwidth?
  rt-types:bandwidth-ieee-float32
+--ro threshold-accelerated-advertisement
  |  +--ro unidirectional-delay?               uint32
  |  +--ro unidirectional-min-delay?           uint32
  |  +--ro unidirectional-max-delay?           uint32
  |  +--ro unidirectional-delay-variation?      uint32
  |  +--ro unidirectional-packet-loss?         decimal64
  |  +--ro unidirectional-residual-bandwidth?
  rt-types:bandwidth-ieee-float32
+--ro unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
+--ro unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
+--rw performance-metric
  |  +--rw unidirectional-delay?               uint32
  |  +--rw unidirectional-min-delay?           uint32
  |  +--rw unidirectional-max-delay?           uint32
  |  +--rw unidirectional-delay-variation?      uint32
  |  +--rw unidirectional-packet-loss?         decimal64
| +--rw unidirectional-residual-bandwidth?
|   rt-types:bandwidth-ieee-float32
| +--rw unidirectional-available-bandwidth?
|   rt-types:bandwidth-ieee-float32
| +--rw unidirectional-utilized-bandwidth?
|   rt-types:bandwidth-ieee-float32
++--rw normality
|   +--rw unidirectional-delay?
|   te-types:performance-metric-normality
|   +--rw unidirectional-min-delay?
|   te-types:performance-metric-normality
|   +--rw unidirectional-max-delay?
|   te-types:performance-metric-normality
|   +--rw unidirectional-delay-variation?
|   te-types:performance-metric-normality
|   +--rw unidirectional-packet-loss?
|   te-types:performance-metric-normality
|   +--rw unidirectional-residual-bandwidth?
|   te-types:performance-metric-normality
|   +--rw unidirectional-available-bandwidth?
|   te-types:performance-metric-normality
|   +--rw unidirectional-utilized-bandwidth?
|   te-types:performance-metric-normality
++--rw throttle
|   +--rw unidirectional-delay-offset?  uint32
|   +--rw measure-interval?            uint32
|   +--rw advertisement-interval?      uint32
|   +--rw suppression-interval?        uint32
|   +--rw throttle-out
|   |   +--rw unidirectional-delay?      uint32
|   |   +--rw unidirectional-min-delay?  uint32
|   |   +--rw unidirectional-max-delay?  uint32
|   |   +--rw unidirectional-delay-variation?  uint32
|   |   +--rw unidirectional-packet-loss? decimal64
|   |   +--rw unidirectional-residual-bandwidth?
|   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-available-bandwidth?
|   rt-types:bandwidth-ieee-float32
|   +--rw unidirectional-utilized-bandwidth?
|   rt-types:bandwidth-ieee-float32
++--rw threshold-in
|   +--rw unidirectional-delay?        uint32
|   +--rw unidirectional-min-delay?    uint32
|   +--rw unidirectional-max-delay?    uint32
|   +--rw unidirectional-delay-variation?  uint32
|   +--rw unidirectional-packet-loss?  decimal64
|   +--rw unidirectional-residual-bandwidth?
|   rt-types:bandwidth-ieee-float32

|  +--rw unidirectional-available-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw unidirectional-utilized-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw threshold-accelerated-advertisement
|     +--rw unidirectional-delay?  uint32
|     +--rw unidirectional-min-delay?  uint32
|     +--rw unidirectional-max-delay?  uint32
|     +--rw unidirectional-delay-variation?  uint32
|     +--rw unidirectional-packet-loss?  decimal64
|     +--rw unidirectional-residual-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw unidirectional-available-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw unidirectional-utilized-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw performance-metric
|     +--rw measurement
|        |  +--rw unidirectional-delay?  uint32
|        |  +--rw unidirectional-min-delay?  uint32
|        |  +--rw unidirectional-max-delay?  uint32
|        |  +--rw unidirectional-delay-variation?  uint32
|        |  +--rw unidirectional-packet-loss?  decimal64
|        |  +--rw unidirectional-residual-bandwidth?  rt-types:bandwidth-ieee-float32
|  +--rw normality
|     |  +--rw unidirectional-delay?  te-types:performance-metric-normality
|     |  +--rw unidirectional-min-delay?  te-types:performance-metric-normality
|     |  +--rw unidirectional-max-delay?  te-types:performance-metric-normality
|     |  +--rw unidirectional-delay-variation?  te-types:performance-metric-normality
|     |  +--rw unidirectional-residual-bandwidth?  te-types:performance-metric-normality
|     |  +--rw unidirectional-available-bandwidth?  te-types:performance-metric-normality
|     |  +--rw unidirectional-utilized-bandwidth?  te-types:performance-metric-normality
te-types:performance-metric-normality
  +--rw throttle
    +--rw unidirectional-delay-offset? uint32
    +--rw measure-interval? uint32
    +--rw advertisement-interval? uint32
    +--rw suppression-interval? uint32
    +--rw threshold-out
      |  +--rw unidirectional-delay? uint32
      |  +--rw unidirectional-min-delay? uint32
      |  +--rw unidirectional-max-delay? uint32
      |  +--rw unidirectional-delay-variation? uint32
      |  +--rw unidirectional-packet-loss? decimal64
      |  +--rw unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-available-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw threshold-in
      |  +--rw unidirectional-delay? uint32
      |  +--rw unidirectional-min-delay? uint32
      |  +--rw unidirectional-max-delay? uint32
      |  +--rw unidirectional-delay-variation? uint32
      |  +--rw unidirectional-packet-loss? decimal64
      |  +--rw unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-available-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw threshold-accelerated-advertisement
      |  +--rw unidirectional-delay? uint32
      |  +--rw unidirectional-min-delay? uint32
      |  +--rw unidirectional-max-delay? uint32
      |  +--rw unidirectional-delay-variation? uint32
      |  +--rw unidirectional-packet-loss? decimal64
      |  +--rw unidirectional-residual-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-available-bandwidth? rt-types:bandwidth-ieee-float32
    +--rw unidirectional-utilized-bandwidth? rt-types:bandwidth-ieee-float32
      +--rw performance-metric
        +--rw measurement
          |  +--rw unidirectional-delay? uint32
          |  +--rw unidirectional-min-delay? uint32
Internet-Draft             YANG L3 TE Topology              October 2017

| +--rw unidirectional-max-delay?             uint32
| +--rw unidirectional-delay-variation?       uint32
| +--rw unidirectional-packet-loss?           decimal64
| +--rw unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
| +--rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
| +--rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32

+-rw normality
| +--rw unidirectional-delay?
te-types:performance-metric-normality
| +--rw unidirectional-min-delay?
te-types:performance-metric-normality
| +--rw unidirectional-max-delay?
te-types:performance-metric-normality
| +--rw unidirectional-delay-variation?
te-types:performance-metric-normality
| +--rw unidirectional-packet-loss?
te-types:performance-metric-normality
| +--rw unidirectional-residual-bandwidth?
te-types:performance-metric-normality
| +--rw unidirectional-available-bandwidth?
te-types:performance-metric-normality
| +--rw unidirectional-utilized-bandwidth?
te-types:performance-metric-normality

+-rw throttle
| +--rw unidirectional-delay-offset?           uint32
| +--rw measure-interval?                      uint32
| +--rw advertisement-interval?                uint32
| +--rw suppression-interval?                  uint32
| +--rw threshold-out
| | +--rw unidirectional-delay?                 uint32
| | +--rw unidirectional-min-delay?             uint32
| | +--rw unidirectional-max-delay?             uint32
| | +--rw unidirectional-delay-variation?       uint32
| | +--rw unidirectional-packet-loss?           decimal64
| | +--rw unidirectional-residual-bandwidth?

types:bandwidth-ieee-float32

| +--rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
| +--rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32

+-rw threshold-in
| +--rw unidirectional-delay?                 uint32
| +--rw unidirectional-min-delay?             uint32
| +--rw unidirectional-max-delay?             uint32
| +--rw unidirectional-delay-variation?       uint32
|  +--rw unidirectional-packet-loss?    decimal64
|  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
|  +--rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
|  +--rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
  +--rw threshold-accelerated-advertisement
  |  +--rw unidirectional-delay?    uint32
  |  +--rw unidirectional-min-delay?  uint32
  |  +--rw unidirectional-max-delay?  uint32
  |  +--rw unidirectional-delay-variation?  uint32
  |  +--rw unidirectional-packet-loss?    decimal64
  |  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
  +--rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
  +--rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
augment /nd:networks/nd:network/lnk:link/tet:te/tet:te-link-attributes:
  +--rw performance-metric
  |  +--rw measurement
  |   |  +--rw unidirectional-delay?    uint32
  |   |  +--rw unidirectional-min-delay?  uint32
  |   |  +--rw unidirectional-max-delay?  uint32
  |   |  +--rw unidirectional-delay-variation?  uint32
  |   |  +--rw unidirectional-packet-loss?    decimal64
  |   |  +--rw unidirectional-residual-bandwidth?
rt-types:bandwidth-ieee-float32
  |  +--rw unidirectional-available-bandwidth?
rt-types:bandwidth-ieee-float32
  |  +--rw unidirectional-utilized-bandwidth?
rt-types:bandwidth-ieee-float32
  +--rw normality
  |  +--rw unidirectional-delay?
types:performance-metric-normality
  |  +--rw unidirectional-min-delay?
types:performance-metric-normality
  |  +--rw unidirectional-max-delay?
types:performance-metric-normality
  |  +--rw unidirectional-delay-variation?
types:performance-metric-normality
  |  +--rw unidirectional-packet-loss?
types:performance-metric-normality
  |  +--rw unidirectional-residual-bandwidth?
types:performance-metric-normality
  |  +--rw unidirectional-available-bandwidth?
te-types:performance-metric-normality
  +--rw unidirectional-utilized-bandwidth?

| te-types:performance-metric-normality
  +--rw throttle
    +-rw unidirectional-delay-offset?      uint32
    +-rw measure-interval?                 uint32
    +-rw advertisement-interval?           uint32
    +-rw suppression-interval?             uint32
    +-rw threshold-out
      +-rw unidirectional-delay?           uint32
      +-rw unidirectional-min-delay?        uint32
      +-rw unidirectional-max-delay?        uint32
      +-rw unidirectional-delay-variation?  uint32
      +-rw unidirectional-packet-loss?      decimal64
      +-rw unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
  +-rw unidirectional-available-bandwidth?

rt-types:bandwidth-ieee-float32
  +-rw unidirectional-utilized-bandwidth?

rt-types:bandwidth-ieee-float32
  +--rw threshold-in
    | +-rw unidirectional-delay?           uint32
    | +-rw unidirectional-min-delay?        uint32
    | +-rw unidirectional-max-delay?        uint32
    | +-rw unidirectional-delay-variation?  uint32
    | +-rw unidirectional-packet-loss?      decimal64
    | +-rw unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
  +-rw unidirectional-available-bandwidth?

rt-types:bandwidth-ieee-float32
  +-rw unidirectional-available-bandwidth?

rt-types:bandwidth-ieee-float32
  +--rw threshold-accelerated-advertisement
    | +-rw unidirectional-delay?           uint32
    | +-rw unidirectional-min-delay?        uint32
    | +-rw unidirectional-max-delay?        uint32
    | +-rw unidirectional-delay-variation?  uint32
    | +-rw unidirectional-packet-loss?      decimal64
    | +-rw unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
  +--rw unidirectional-available-bandwidth?

rt-types:bandwidth-ieee-float32
  +--rw unidirectional-available-bandwidth?

rt-types:bandwidth-ieee-float32
  +--rw unidirectional-available-bandwidth?

augment /nd:networks/nd:network/lnk:link/tet:te
/td:information-source-entry:
  +-ro performance-metric
  +-ro measurement
| +--ro unidirectional-delay?                uint32
| +--ro unidirectional-min-delay?            uint32
| +--ro unidirectional-max-delay?            uint32
| +--ro unidirectional-delay-variation?      uint32
| +--ro unidirectional-packet-loss?          decimal64
| +--ro unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
| +--ro unidirectional-available-bandwidth?
| +--ro unidirectional-utilized-bandwidth?

rt-types:bandwidth-ieee-float32
| +--ro normality
| +--ro unidirectional-delay?
te-types:performance-metric-normality
| +--ro unidirectional-min-delay?
te-types:performance-metric-normality
| +--ro unidirectional-max-delay?
te-types:performance-metric-normality
| +--ro unidirectional-delay-variation?
te-types:performance-metric-normality
| +--ro unidirectional-packet-loss?
te-types:performance-metric-normality
| +--ro unidirectional-residual-bandwidth?
te-types:performance-metric-normality
| +--ro unidirectional-available-bandwidth?
te-types:performance-metric-normality
| +--ro unidirectional-utilized-bandwidth?
te-types:performance-metric-normality

++--ro throttle
| +--ro unidirectional-delay-offset?         uint32
| +--ro measure-interval?                   uint32
| +--ro advertisement-interval?             uint32
| +--ro suppression-interval?               uint32
| +--ro threshold-out
| | +--ro unidirectional-delay?               uint32
| | +--ro unidirectional-min-delay?           uint32
| | +--ro unidirectional-max-delay?           uint32
| | +--ro unidirectional-delay-variation?     uint32
| | +--ro unidirectional-packet-loss?         decimal64
| | +--ro unidirectional-residual-bandwidth?

rt-types:bandwidth-ieee-float32
| +--ro unidirectional-available-bandwidth?
| +--ro unidirectional-utilized-bandwidth?

rt-types:bandwidth-ieee-float32
| +--ro threshold-in
| | +--ro unidirectional-delay?               uint32
| | +--ro unidirectional-min-delay?           uint32

4.  YANG Modules

4.1.  Layer 3 TE Topology Module

<CODE BEGINS> file "ietf-l3-te-topology@2017-07-03.yang"
module ietf-l3-te-topology {
  yang-version 1.1;
  prefix "l3tet";

import ietf-network {
   prefix "nd";
}
import ietf-network-topology {
   prefix "lnk";
}
import ietf-l3-unicast-topology {
   prefix "l3t";
}
import ietf-te-topology {
   prefix "tet";
}

organization "TBD";
contact "TBD";
description "L3 TE Topology model";

revision 2017-07-03 {
   description "Initial revision";
   reference "TBD";
}

grouping l3-te-topology-type {
   description
      "Identifies the L3 TE topology type.";
   container l3-te {
      presence "indicates L3 TE Topology";
      description
         "Its presence identifies the L3 TE topology type.";
   }
}

augment "/nd:nets/nd:network/nd:network-types/"
+ "l3t:l3-unicast-topology" {
   description
      "Defines the L3 TE topology type.";
   uses l3-te-topology-type;
}

augment "/nd:nets/nd:network/l3t:l3-topology-attributes" {
   when "./nd:network-types/l3t:l3-unicast-topology/l3-te" {
      description "Augment only for L3 TE topology";
   }
   description "Augment topology configuration";
   uses l3-te-topology-attributes;
}
augment "/nd:networks/nd:network/nd:node/l3t:l3-node-attributes" {
  when "../../nd:network-types/l3t:l3-unicast-topology/l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment node configuration";
  uses l3-te-node-attributes;
}

augment "/nd:networks/nd:network/nd:node/lnk:termination-point/" + "l3t:l3-termination-point-attributes" {
  when "../../../nd:network-types/l3t:l3-unicast-topology/" + "l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment termination point configuration";
  uses l3-te-tp-attributes;
}

augment "/nd:networks/nd:network/lnk:link/l3t:l3-link-attributes" {
  when "../../../nd:network-types/l3t:l3-unicast-topology/l3-te" {
    description "Augment only for L3 TE topology";
  }
  description "Augment link configuration";
  uses l3-te-link-attributes;
}

grouping l3-te-topology-attributes {
  description "L3 TE topology scope attributes";
  container l3-te-topology-attributes {
    must "/nd:networks/nd:network" + 
    "[nd:network-id = current()/network-ref]/nd:network-types/" + 
    "tet:te-topology" {
      error-message
      "The referenced network must be a TE topology.";
      description
      "The referenced network must be a TE topology.";
    }
    description "Containing TE topology references";
    uses nd:network-ref;
  } // l3-te-topology-attributes
} // l3-te-topology-attributes

grouping l3-te-node-attributes {
  description "L3 TE node scope attributes";
  container l3-te-node-attributes {
    must "/nd:networks/nd:network" + 
    "[nd:network-id = current()/network-ref]/nd:network-types/" + 
    "tet:te-topology" {
    }
  } // l3-te-node-attributes
} // l3-te-node-attributes
error-message
  "The referenced network must be a TE topology.";
description
  "The referenced network must be a TE topology.";
}
description "Containing TE node references";
uses nd:node-ref;
} // l3-te
} // l3-te-node-attributes

grouping l3-te-tp-attributes {
  description "L3 TE termination point scope attributes"
  container l3-te-tp-attributes {
    must "/nd:networks/nd:network"
    + 
      "[nd:network-id = current()/network-ref]/nd:network-types/
      + 
        "tet:te-topology" {
      error-message
        "The referenced network must be a TE topology.";
description
        "The referenced network must be a TE topology.";
    }
description "Containing TE termination point references";
    uses lnk:tp-ref;
  } // l3-te
} // l3-te-tp-attributes

grouping l3-te-link-attributes {
  description "L3 TE link scope attributes"
  container l3-te-link-attributes {
    must "/nd:networks/nd:network"
    + 
      "[nd:network-id = current()/network-ref]/nd:network-types/
      + 
        "tet:te-topology" {
      error-message
        "The referenced network must be a TE topology.";
description
        "The referenced network must be a TE topology.";
    }
description "Containing TE link references";
    uses lnk:link-ref;
  } // l3-te
} // l3-te-link-attributes
} // l3-te
4.2. Packet Switching TE Topology Module

<CODE BEGINS> file "ietf-te-topology-packet@2017-10-29.yang"
module ietf-te-topology-packet {
    yang-version 1;
    prefix "tet-pkt";

    import ietf-network {
        prefix "nd";
    }

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

    import ietf-routing-types {
        prefix "rt-types";
    }

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

    import ietf-te-types {
        prefix "te-types";
    }

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

    contact
        *WG Web: <http://tools.ietf.org/wg/teas/> *
        WG List: <mailto:teas@ietf.org>
        WG Chair: Lou Berger
            <mailto:lberger@labn.net>
        WG Chair: Vishnu Pavan Beeram
            <mailto:vbeeram@juniper.net>
        Editors: Xufeng Liu
            <mailto:Xufeng_Liu@jabil.com>
            Igor Bryskin
description "TE topology model";

revision 2017-10-29 {
  description "Initial revision";
  reference "TBD";
}

/*
 * Features
 */

feature te-performance-metric {
  description "This feature indicates that the system supports
  TE performance metric.";
  RFC7823: Performance-Based Path Selection for Explicitly
  Routed Label Switched Paths (LSPs) Using TE Metric
  Extensions";
}

/*
 * Groupings
 */

grouping packet-switch-capable-container {
  description "The container of packet switch capable attributes.";
  container packet-switch-capable {
    description "Interface has packet-switching capabilities.";
    leaf minimum-lsp-bandwidth {
      type rt-types:bandwidth-ieee-float32;
    }
  }
}
description
"Minimum LSP Bandwidth. Units in bytes per second";
}
leaf interface-mtu {
    type uint16;
    description
"Interface MTU.";
}

/
* Augmentations
/*
/* Augmentations to connectivity-matrix */
+ "tet:te-node-attributes/tet:connectivity-matrices" {
    description
"Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}
+ "tet:te-node-attributes/tet:connectivity-matrices/"
+ "tet:connectivity-matrix" {
    description
"Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}
}
+ "tet:information-source-entry/tet:connectivity-matrices" {
    description
"Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}
}
+ "tet:information-source-entry/tet:connectivity-matrices/"
+ "tet:connectivity-matrix" {
description
  "Parameters for PSC TE topology."
uses te-types:performance-metric-container {
  if-feature te-performance-metric;
}

/* Augmentations to tunnel-termination-point */
  + "tet:tunnel-termination-point/"
  + "tet:local-link-connectivities" {
    description
    "Parameters for PSC TE topology."
    uses te-types:performance-metric-container {
      if-feature te-performance-metric;
    }
  }

  + "tet:tunnel-termination-point/"
  + "tet:local-link-connectivities/"
  + "tet:local-link-connectivity" {
    description
    "Parameters for PSC TE topology."
    uses te-types:performance-metric-container {
      if-feature te-performance-metric;
    }
  }

/* Augmentations to te-link-attributes */
augment "/nd:networks/tet:te/tet:templates/
  + "tet:link-template/tet:te-link-attributes" {
    when "tet:interface-switching-capability "
      + ":[tet:switching-capability = ‘te-types:switching-pscl’]"
      {
        description "Valid only for PSC";
      }
    description
    "Parameters for PSC TE topology."
    uses te-types:performance-metric-container {
      if-feature te-performance-metric;
    }
  }

  + "tet:te-link-attributes" {
    when "tet:interface-switching-capability "
      + "[tet:switching-capability = ‘te-types:switching-pscl’]"
      {
        description "Valid only for PSC";
      }
    description
    "Parameters for PSC TE topology."
    uses te-types:performance-metric-container {
      if-feature te-performance-metric;
    }
  }
augment "/nd:networks/nd:network/lnk:link/tet:te/"
+ "tet:te-link-attributes/"
    + "tet:interface-switching-capability"
    when "tet:switching-capability = ’te-types:switching-psc1’" {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology."
    uses te-types:performance-metric-container {
        if-feature te-performance-metric;
    }
}

/* Augmentations to interface-switching-capability */
augment "/nd:networks/tet:te/tet:templates/"
    + "tet:link-template/tet:te-link-attributes/"
    + "tet:interface-switching-capability"
    when "tet:switching-capability = ’te-types:switching-psc1’" {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology."
    uses packet-switch-capable-container;
}

augment "/nd:networks/nd:network/lnk:link/tet:te/"
    + "tet:link-attributes/"
    + "tet:interface-switching-capability"
    when "tet:switching-capability = ’te-types:switching-psc1’" {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology."
    uses packet-switch-capable-container;
}

augment "/nd:networks/nd:network/lnk:link/tet:te/"
    + "tet:information-source-entry/"
    + "tet:interface-switching-capability"
    when "tet:switching-capability = ’te-types:switching-psc1’" {
        description "Valid only for PSC";
    }
    description
        "Parameters for PSC TE topology."
    uses packet-switch-capable-container;
}
5. IANA Considerations

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

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

--------------------------------------------------------------------
Registnt Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------

--------------------------------------------------------------------
Registnt Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------

--------------------------------------------------------------------
Registnt Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------

--------------------------------------------------------------------
Registnt Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

6. Security Considerations

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

7. References

7.1. Normative References

7.2. Informative References

[I-D.ietf-i2rs-yang-l3-topology]

[I-D.ietf-teas-yang-te-topo]
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG modules ietf-l3-te-topology and ietf-te-topology-packet defined in this document are designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [I-D.ietf-netmod-revised-datastores]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion modules, ietf-l3-te-topology-state and ietf-te-topology-packet-state, are defined as state models, which mirror the modules ietf-l3-te-topology and ietf-te-topology-packet defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion modules, ietf-l3-te-topology-state and ietf-te-topology-packet-state, are redundant and SHOULD NOT be supported by implementations that support NMDA.

As the structure of the companion modules mirrors that of the corresponding NMDA models, the YANG trees of the companion modules are not depicted separately.

A.1. Layer 3 TE Topology State Module

```yang
<CODE BEGINS> file "ietf-l3-te-topology-state@2017-07-03.yang"
module ietf-l3-te-topology-state {
  yang-version 1.1;
  prefix "l3tet-s";
  
  import ietf-l3-te-topology {
    prefix "l3tet";
  }
  import ietf-network-state {
    prefix "nd-s";
  }
  import ietf-network-topology-state {
    prefix "lnk-s";
  }
  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
  }

  organization "TBD";
  contact "TBD";
  description "L3 TE Topology model";
</CODE BEGINS>
```
revision 2017-07-03 {
    description "Initial revision";
    reference "TBD";
}

augment "/nd-s:networks/nd-s:network/nd-s:network-types/"
+ "l3t-s:l3-unicast-topology" {
    description
    "Defines the L3 TE topology type.";
    uses l3tet:l3-te-topology-type;
}

augment "/nd-s:networks/nd-s:network/"
+ "l3t-s:l3-topology-attributes" {
    when "../nd-s:network-types/l3t-s:l3-unicast-topology/l3-te" {
        description "Augment only for L3 TE topology";
    }
    description "Augment topology configuration";
    uses l3tet:l3-te-topology-attributes;
}

augment "/nd-s:networks/nd-s:network/nd-s:node/"
+ "l3t-s:l3-node-attributes" {
    when "../../nd-s:network-types/l3t-s:l3-unicast-topology/l3-te" {
        description "Augment only for L3 TE topology";
    }
    description "Augment node configuration";
    uses l3tet:l3-te-node-attributes;
}

augment "/nd-s:networks/nd-s:network/nd-s:node/termination-point/"
+ "l3t-s:l3-termination-point-attributes" {
    when "../../../nd-s:network-types/l3t-s:l3-unicast-topology/"
+ "l3-te" {
        description "Augment only for L3 TE topology";
    }
    description "Augment termination point configuration";
    uses l3tet:l3-te-tp-attributes;
}

augment "/nd-s:networks/nd-s:network/lnk-s:link/"
+ "l3t-s:l3-link-attributes" {
    when "../../nd-s:network-types/l3t-s:l3-unicast-topology/l3-te" {
        description "Augment only for L3 TE topology";
    }
    description "Augment link configuration";
uses l3tet:l3-te-link-attributes;
}
}<CODE ENDS>

A.2. Packet Switching TE Topology State Module

<CODE BEGINS> file "ietf-te-topology-packet-state@2017-10-29.yang"
module ietf-te-topology-packet-state {
  yang-version 1;
  namespace
  prefix "tet-pkt-s";
  import ietf-te-topology-packet {
    prefix "tet-pkt";
  }
  import ietf-network-state {
    prefix "nd-s";
  }
  import ietf-network-topology-state {
    prefix "lnk-s";
  }
  import ietf-te-topology-state {
    prefix "tet-s";
  }
  import ietf-te-types {
    prefix "te-types";
  }
  organization
    "Traffic Engineering Architecture and Signaling (TEAS)
    Working Group";
  contact
    "WG Web:  <http://tools.ietf.org/wg/teas/>
    WG List:  <mailto:teas@ietf.org>
    WG Chair: Lou Berger
    <mailto:lberger@labn.net>

description "TE topology model";

revision 2017-10-29 {
    description "Initial revision";
    reference "TBD";
}

/*
* Augmentations
*/
/* Augmentations to connectivity-matrix */
augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:te-node-attributes/tet-s:connectivity-matrices" {
        description "Parameters for PSC TE topology.";
        uses te-types:performance-metric-container {
            if-feature tet-pkt:te-performance-metric;
        }
    }

augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:te-node-attributes/tet-s:connectivity-matrices/"
    + "tet-s:connectivity-matrix" {
        description "Parameters for PSC TE topology.";
        uses te-types:performance-metric-container {
            if-feature tet-pkt:te-performance-metric;
        }
    }
augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:information-source-entry/
    + "tet-s:connectivity-matrices" {
    description
    "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature tet-pkt:te-performance-metric;
    }
}

augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:information-source-entry/
    + "tet-s:connectivity-matrices/
    + "tet-s:connectivity-matrix" {
    description
    "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature tet-pkt:te-performance-metric;
    }
}

/* Augmentations to tunnel-termination-point */
augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:tunnel-termination-point/
    + "tet-s:local-link-connectivities" {
    description
    "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature tet-pkt:te-performance-metric;
    }
}

augment "/nd-s:networks/nd-s:network/nd-s:node/tet-s:te/
    + "tet-s:tunnel-termination-point/
    + "tet-s:local-link-connectivities/
    + "tet-s:local-link-connectivity" {
    description
    "Parameters for PSC TE topology.";
    uses te-types:performance-metric-container {
        if-feature tet-pkt:te-performance-metric;
    }
}

/* Augmentations to te-link-attributes */
augment "/nd-s:networks/tet-s:te/tet-s:templates/"
+ "tet-s:link-template/tet-s:te-link-attributes" {
  when "tet-s:interface-switching-capability"
  + "[tet-s:switching-capability = 'te-types:switching-pscl']"
    description "Valid only for PSC";
}

description
  "Parameters for PSC TE topology.";
uses te-types:performance-metric-container {
  if-feature tet-pkt:te-performance-metric;
}
}

augment "/nd-s:nets/nd-s:network/lnk-s:link/tet-s:te/"
  + "tet-s:te-link-attributes" {
  when "tet-s:interface-switching-capability"
  + "[tet-s:switching-capability = 'te-types:switching-pscl']"
    description "Valid only for PSC";
}

description
  "Parameters for PSC TE topology.";
uses te-types:performance-metric-container {
  if-feature tet-pkt:te-performance-metric;
}
}

/* Augmentations to interface-switching-capability */
augment "/nd-s:nets/tet-s:te/tet-s:templates/"
  + "tet-s:link-template/tet-s:te-link-attributes/"
  + "tet-s:interface-switching-capability"
    when "tet-s:switching-capability = 'te-types:switching-pscl'"
    description "Valid only for PSC";
}

description
  "Parameters for PSC TE topology.";
uses tet-pkt:packet-switch-capable-container;

augment "/nd-s:networks/nd-s:network/lnk-s:link/tet-s:te/"
   + "tet-s:te-link-attributes/
   + "tet-s:interface-switching-capability" {
   when "tet-s:switching-capability = 'te-types:switching-psc1' " {
         description "Valid only for PSC";
   }
   description
   "Parameters for PSC TE topology.";
   uses tet-pkt:packet-switch-capable-container;
}

augment "/nd-s:networks/nd-s:network/lnk-s:link/tet-s:te/"
   + "tet-s:information-source-entry/
   + "tet-s:interface-switching-capability" {
   when "tet-s:switching-capability = 'te-types:switching-psc1' " {
         description "Valid only for PSC";
   }
   description
   "Parameters for PSC TE topology.";
   uses tet-pkt:packet-switch-capable-container;
}

<CODE ENDS>

Authors’ Addresses

Xufeng Liu
Jabil
8281 Greensboro Drive, Suite 200
McLean VA 22102
USA

EMail: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei Technologies

EMail: Igor.Bryskin@huawei.com

YANG Data Model for SR and SR TE Topologies
draft-liu-teas-yang-sr-te-topo-04

Abstract

This document defines a YANG data model for Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at 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 3, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of
1. Introduction

This document defines a YANG [RFC7950] data model for describing the presentations of Segment Routing (SR) topology and Segment Routing (SR) traffic engineering (TE) topology.

1.1. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119].

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

- augment
- data model
2. Modeling Considerations

2.1. Segment Routing (SR) topology

The Layer 3 network topology model is discussed in [I-D.ietf-i2rs-yang-l3-topology]. The Segment Routing (SR) topology model proposed in this document augments and uses the ietf-l3-unicast-igp-topology module defined in [I-D.ietf-i2rs-yang-l3-topology]. SR related attributes are covered in the ietf-sr-topology model.

```
+-------------------------------+                  +-------------------------------+
| Layer 3 Network Topology     |  v              | L3 TE Topology                |
| ietf-l3-unicast-topology     |                  | ietf-l3-te-topology           |
+-------------------------------+                  +-------------------------------+
                  \                  /\                  /
                  \                  / \                  /
                   v                  v                   v
+-------------------------------+                  +-------------------------------+
| SR Topology                   |                  | SR TE Topology                |
| ietf-sr-topology              |                  |                               |
+-------------------------------+                  +-------------------------------+
```

2.2. Segment Routing (SR) TE topology

When traffic engineering is enabled on an SR topology, there will be associations between objects in SR topologies and objects in TE topologies. An SR TE topology is both an SR topology and a layer 3 TE topology. Multiple inheritance is used to achieve such relations.

```
+-------------------------------+                  +-------------------------------+
| SR Topology                   |  v              | L3 TE Topology                |
| ietf-sr-topology              |                  | ietf-l3-te-topology           |
+-------------------------------+                  +-------------------------------+
                  \                  /\                  /
                  \                  / \                  /
                   v                  v                   v
+-------------------------------+                  +-------------------------------+
| SR TE Topology                |                  |                               |
+-------------------------------+                  +-------------------------------+
```
Each type of topologies is indicated by "network-types" defined in [I-D.ietf-i2rs-yang-network-topo]. For the three types of topologies above, the data representations are:

L3 Topology:
/nd:networks/nd:network/nd:network-types/l3-unicast-topology

L3 TE Topology:
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te

SR Topology:

SR TE Topology: (multiple inheritance)
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te
/nd:networks/nd:network/nd:network-types/l3-unicast-topology/l3-te

2.3. Relations to ietf-segment-routing

[I-D.ietf-spring-sr-yang] defines ietf-segment-routing that is a model intended to be used on network elements to configure or operate segment routing; ietf-sr-topology defined in this document is intended to be used on a controller for the network-wide operations such as path computation.

SR topology model shares many modeling constructs defined in ietf-segment-routing. The module ietf-sr-topology uses the types and groupings defined in ietf-segment-routing.

2.4. Open Items

a. Protection on link: The feature of link protection will be modeled in the next revision.

b. Link bundle: The feature of link bundle will be modeled in the next revision.

3. Model Structure

The model tree structure of the Segment Routing (SR) topology module is as shown below:
module: ietf-sr-topology

augment /nw:networks/nw:network/nw:network-types
/l3t:13-unicast-topology:
  +--rw sr!

augment /nw:networks/nw:network/l3t:13-topology-attributes:
  +--rw sr
    +--rw srgb* [lower-bound upper-bound]
    |  +--rw lower-bound uint32
    |  +--rw upper-bound uint32

augment /nw:networks/nw:network/nw:node/l3t:13-node-attributes:
  +--rw sr
    +--rw srgb* [lower-bound upper-bound]
    |  +--rw lower-bound uint32
    |  +--rw upper-bound uint32
    +--rw sr.lb* [lower-bound upper-bound]
    |  +--rw lower-bound uint32
    |  +--rw upper-bound uint32
    +--rw node-capabilities
    |  +--rw transport-planes* [transport-plane]
    |  |  +--rw transport-plane identityref
    |  +--rw readable-label-stack-depth? uint8
    +--ro information-source? enumeration
    +--ro information-source-state
    +--ro credibility-preference? uint16

augment /nw:networks/nw:network/nw:node/l3t:13-node-attributes
/l3t:prefix:
  +--rw sr!
    +--rw value-type? enumeration
    +--rw start-sid uint32
    +--rw range? uint32
    +--rw algorithm? identityref
    +--rw last-hop-behavior? {sid-last-hop-behavior}?
    +--rw is-local? boolean

augment /nw:networks/nw:network/nw:node/nt:termination-point
/l3t:13-termination-point-attributes:
  augment /nw:networks/nw:network/nt:link/l3t:13-link-attributes:
  +--rw sr
    +--rw sid? uint32
    +--rw value-type? enumeration
    +--rw is-local? boolean
    +--ro is-part-of-set? boolean
    +--ro is-on-lan? boolean
    +--ro information-source? enumeration
    +--ro information-source-state
    +--ro credibility-preference? uint16
4. YANG Module

<CODE BEGINS> file "ietf-sr-topology@2017-10-30.yang"
module ietf-sr-topology {
    yang-version 1.1;
    prefix "srt";

    import ietf-network {
        prefix "nw";
    }
    import ietf-network-topology {
        prefix "nt";
    }
    import ietf-l3-unicast-topology {
        prefix "l3t";
    }
    import ietf-segment-routing-common {
        prefix "sr-cmn";
    }

    organization "TBD";
    contact "TBD";
    description "L3 TE Topology model";

    revision 2017-10-30 {
        description "Initial revision";
        reference "TBD";
    }

    grouping sr-topology-type {
        description
            "Identifies the SR topology type.";
        container sr {
            presence "Indicates SR Topology";
            description
                "Its presence identifies the SR topology type.";
        }
    }

    augment "/nw:networks/nw:network/nw:network-types/"
        + "l3t:l3-unicast-topology" {
        description
            "Defines the SR topology type.";
        uses sr-topology-type;
    }

augment "/nw:networks/nw:network/l3t:13-topology-attributes" {
  when "../nw:network-types/l3t:13-unicast-topology/sr" {
    description "Augment only for SR topology.";
  }
  description "Augment topology configuration";
  uses sr-topology-attributes;
}

augment "/nw:networks/nw:network/nw:node/l3t:13-node-attributes" {
  when "../../nw:network-types/l3t:13-unicast-topology/sr" {
    description "Augment only for SR topology.";
  }
  description "Augment node configuration.";
  uses sr-node-attributes;
}

augment "/nw:networks/nw:network/nw:node/l3t:13-node-attributes" + "/l3t:prefix" {
  when "../../../nw:network-types/l3t:13-unicast-topology/sr" {
    description "Augment only for SR topology.";
  }
  description "Augment node prefix.";
  uses sr-node-prefix-attributes;
}

augment "/nw:networks/nw:network/nw:node/nt:termination-point/" + "/l3t:termination-point-attributes" {
  when "../../../../nw:network-types/l3t:13-unicast-topology/" + "/sr" {
    description "Augment only for SR topology.";
  }
  description "Augment termination point configuration";
  uses sr-tp-attributes;
}

augment "/nw:networks/nw:network/nt:link/l3t:13-link-attributes" {
  when "../../nw:network-types/l3t:13-unicast-topology/sr" {
    description "Augment only for SR topology.";
  }
  description "Augment link configuration.";
  uses sr-link-attributes;
}

grouping sr-topology-attributes {
  description "SR topology scope attributes.";
  container sr {
    description "Containing SR attributes.";
  }
}
uses sr-cmn:srgb-cfg;
} // sr
} // sr-topology-attributes

grouping information-source-attributes {
    description
        "The attributes identifying source that has provided the related information, and the source credibility.";

    leaf information-source {
        type enumeration {
            enum "unknown" {
                description "The source is unknown.";
            }
            enum "locally-configured" {
                description "Configured entity.";
            }
            enum "ospfv2" {
                description "OSPFv2.";
            }
            enum "ospfv3" {
                description "OSPFv3.";
            }
            enum "isis" {
                description "ISIS.";
            }
            enum "system-processed" {
                description "System processed entity.";
            }
            enum "other" {
                description "Other source.";
            }
        }
        config false;
        description
            "Indicates the source of the information.";
    }

    container information-source-state {
        config false;
        description
            "The container contains state attributes related to the information source.";

        leaf credibility-preference {
            type uint16;
            description
                "The preference value to calculate the traffic engineering database credibility value used for tie-break selection between different information-source values.";
        }
    }
}
Higher value is more preferable.;
uses sr-cmn:sid-value-type;
leaf is-local {
  type boolean;
  description
    "true' if the SID is local.";
}
leaf is-part-of-set {
  type boolean;
  config false;
  description
    "true' if the SID is part of a set.";
}
leaf is-on-lan {
  type boolean;
  config false;
  description
    "true' if on a lan.";
}
uses information-source-attributes;
} // sr
} // sr-link-attributes

5. IANA Considerations

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

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

------------------------------------------
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
------------------------------------------

------------------------------------------
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
------------------------------------------

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

6. Security Considerations

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

7. References

7.1. Normative References


7.2. Informative References

[I-D.ietf-i2rs-yang-network-topo]
Clemm, A., Medved, J., Varga, R., Bahadur, N.,
Ananthakrishnan, H., and X. Liu, "A Data Model for Network
Topologies", draft-ietf-i2rs-yang-network-topo-17 (work in
progress), October 2017.

[I-D.ietf-i2rs-yang-l3-topology]
Clemm, A., Medved, J., Varga, R., Liu, X.,
Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model
for Layer 3 Topologies", draft-ietf-i2rs-yang-
l3-topology-12 (work in progress), October 2017.

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and
O. Dios, "YANG Data Model for TE Topologies", draft-ietf-

[I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Sarkar, P., and J. Tantsura, "YANG
Data Model for Segment Routing", draft-ietf-spring-sr-
yang-07 (work in progress), July 2017.
Appendix A. Companion YANG Model for Non-NMDA Compliant Implementations

The YANG module ietf-sr-topology defined in this document is designed to be used in conjunction with implementations that support the Network Management Datastore Architecture (NMDA) defined in [I-D.ietf-netmod-revised-datastores]. In order to allow implementations to use the model even in cases when NMDA is not supported, the following companion module, ietf-sr-topology-state, is defined as state model, which mirrors the module ietf-sr-topology defined earlier in this document. However, all data nodes in the companion module are non-configurable, to represent the applied configuration or the derived operational states.

The companion module, ietf-sr-topology-state, is redundant and SHOULD NOT be supported by implementations that support NMDA.

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

A.1. SR Topology State Module

<CODE BEGINS> file "ietf-sr-topology-state@2017-10-30.yang"
module ietf-sr-topology-state {
  yang-version 1.1;
  prefix "srt-s";

  import ietf-sr-topology {
    prefix "srt";
  }
  import ietf-network-state {
    prefix "nw-s";
  }
  import ietf-network-topology-state {
    prefix "nt-s";
  }
  import ietf-l3-unicast-topology-state {
    prefix "l3t-s";
  }
  import ietf-segment-routing-common {
    prefix "sr-cmn";
  }

  organization "TBD";
  contact "TBD";
  description "L3 TE Topology model";

augment "/nw-s:networks/nw-s:network/nw-s:network-types/"
  + "l3t-s:l3-unicast-topology" {
    description
    "Defines the SR topology type.";
    uses srt:sr-topology-type;
  }

augment "/nw-s:networks/nw-s:network/
  + "l3t-s:l3-topology-attributes" {
    when "../nw-s:network-types/l3t-s:l3-unicast-topology/sr" {
      description "Augment only for SR topology.";
    }
    description "Augment topology configuration";
    uses srt:sr-topology-attributes;
  }

augment "/nw-s:networks/nw-s:network/nw-s:node/
  + "l3t-s:l3-node-attributes" {
    when "../../nw-s:network-types/l3t-s:l3-unicast-topology/sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node configuration.";
    uses srt:sr-node-attributes;
  }

augment "/nw-s:networks/nw-s:network/nw-s:node/
  + "l3t-s:l3-node-attributes/l3t-s:prefix" {
    when "../../../nw-s:network-types/l3t-s:l3-unicast-topology/sr" {
      description "Augment only for SR topology.";
    }
    description "Augment node prefix.";
    uses srt:sr-node-prefix-attributes;
  }

augment "/nw-s:networks/nw-s:network/nw-s:node/
  + "nt-s:termination-point/
    + "l3t-s:l3-termination-point-attributes" {
    when "../../../../nw-s:network-types/l3t-s:l3-unicast-topology/"
      + "sr" {
      description "Augment only for SR topology.";
    }
    description "Augment termination point configuration";
    uses srt:sr-tp-attributes;
augment "*/nw-s:networks/nw-s:network/nt-s:link/"
+ "l3t-s:l3-link-attributes"
  when "../../nw-s:network-types/l3t-s:l3-unicast-topology/sr"
  description "Augment only for SR topology."
  uses srt:sr-link-attributes;

grouping sr-topology-attributes {
  description "SR topology scope attributes."
  container sr {
    description "Containing SR attributes."
    uses sr-cmn:srgb-cfg;
  }
}

Authors' Addresses

Xufeng Liu
Jabil
8281 Greensboro Drive, Suite 200
McLean VA 22102
USA
EMail: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei Technologies
EMail: Igor.Bryskin@huawei.com

Vishnu Pavan Beeram
Juniper Networks
EMail: vbeeram@juniper.net
Controlling pre-standard coherent Optical Interfaces
draft-many-coherent-dwdm-if-control-01

Abstract

Modulated optical interfaces with coherent detection receivers are in widespread use in Internet networking equipment. Various implementations are in deployment since 2012 but there is no standard available defining those interfaces, nor their capabilities. This document identifies the need for work on control plane aspects pre-standard coherent optical DWDM applications.

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 10, 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
1. Introduction

The dominant interconnection technology in the Internet is based on fiber and optical transceiver interfaces. Multiple SDOs are working on control-plane and data-plane standards in this field. Some are covering LAN applications (IEEE), while others work on WAN and in particular DWDM based applications (ITU-T SG15). Those DWDM related recommendations are based on non-coherent detection schemes and do not cover modulated optical signals and coherent detection. DWDM wavelengths of 40Gb/s, 100Gb/s and beyond use higher order modulation techniques with coherent detection schemes and are used throughout the industry. Current implementations are already heading towards capacities of 400Gb/s and 1Tb/s per Interface. The gap between standards availability and practical deployment creates a mounting need in the industry for a common data model that can be used to control Pre Standard Coherent Optical (PRESCO) DWDM interfaces. This document addresses the issue of progressing control plane work related to PRESCO-DWDM technology.
2. Requirements Language

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

3. Abbreviations

- DWDM: Dense Wavelength Division Multiplexing
- PRESCO-interface: PRE-Standard-Coherent-Optical DWDM interface
- PRESCO-wavelength: pre-standard coherent DWDM signal with phase modulated transmitter and coherent detection receiver
- PRESCO-Module: Transmitter/receiver Module of pre-standard coherent DWDM signals
- SDO: Standards Developing Organization

4. Motivation

As data plane standards for standard coherent optical interfaces are in flux, there is a lack of common ground on modeling and encoding interface parameters related to higher order modulation techniques with coherent detection schemes. This unnecessarily burdens control systems with complexity in coping with incompatible implementations and complex operation. As data plane standards are insufficient to provide guidance on control plane work for PRESCO devices, work on data models for PRESCO applications need to proceed independently of data plane standards.

5. Applicability to CCAMP

"The CCAMP working group is responsible for standardizing a common control plane and a separate common measurement plane for non-packet technologies found in the Internet and in the networks of telecom service providers (ISPs and SPs)." As such CCAMP is chartered with "the definition of management objects (e.g., as part of MIB modules or YANG models) and control of OAM techniques relevant to the protocols and extensions specified within the WG" [ccamp-charter]. Hence work to control PRESCO devices is in the scope of ccamp.

6. State of Standards
6.1. Data Plane

The first version of [superseded-ITU.G698.2] was published July 2007. A key role plays the concept of "application codes" to characterize transmitters and receivers using an character-string. When a Tx/Rx pair, with the same code, is connected over a link with specific optical properties, the DWDM connection is guaranteed to interoperate regardless of their origin, i.e. their manufacturer within the conditions set out in this recommendation.

The now in-force [ITU.G698.2] recommendation "includes unidirectional DWDM applications at 2.5 and 10 Gbit/s with 100 GHz channel frequency spacing as well as applications at 10 Gbit/s with 50 GHz channel frequency spacing. No comparable Recommendation exists for higher bitrates and higher modulated signals and the current version of [RFC7581] is based on those codes as well. Also related standards like [ITU.G697] "Optical monitoring for dense wavelength division multiplexing systems" and [ITU.G680] "Physical transfer functions of optical network elements" are written based on 2.5G and 10G technology using direct detection.

ITU-T Study Group 15 set out in 2010 to begin work toward "revision of [SG15-2012], establishing sets of parameters and associated values to enable multi-vendor interoperability for 40 Gbit/s application codes with various modulation formats". At the time this work began, there were several candidate modulation formats e.g DQPSK, PM-QPSK, OFDM-QPSK. NRZ modulation used for 2.5G and 10G is simpler than the so called advanced phase modulation schemes required for higher bitrates. For DP-QPSK, for example, it is necessary to determine and specify suitable parameters for the characterization of the polarization and phase components of the signal. The challenge experts had in advancing that work has been brought to the attention of ccamp in 2013 when it was reported that progress have been made such that "there is only 1 modulation format candidate for 100G standardization" and "what we're struggling with is doing fundamental work on standardizing phase modulated transmission"[IETF-86-ccamp-minutes].

We recognize the complex nature of the task and acknowledge the amount of diligent work that has been put, and is still being put into coherent DWDM data-plane standardization. Still, at the time of this writing, the fundamental underlying problem in standardizing phase modulated optical signals is not yet solved. Even when that work would be completed, the relevant aspects of the link and receiver, together with an appropriate FEC standardization, will need to be addressed as well. By Sept 2016 there is no deadline communicated by when this work is expected to be finalized.
6.2. Control Plane

Standardization control aspects in relevant SDOs naturally follows the work on data plane which provides the base for monitoring capabilities and identification of critical parameters. Available recommendations utilize application codes defined in [ITU.G698.2]. Such application code can be considered a character based abbreviation (such as e.g. DScW-ytz(v) from [ITU.G698.2]) to characterize transceiver characteristics. Consequently control information in WSON (see [RFC3591] and [ITU.G874.1]) is also based on the use of such application codes. [RFC3591] is based on [ITU.G872]. and provides a starting point for the definition and structuring of objects. However, given that [RFC3591] is also a standards track document, it naturally based on standard definitions and can not include parameters required to describe PRESCO devices. In parallel, individual contributions in ccamp (e.g. [draft-2012]) were proposing to work by introducing extensions for parameters describing PRESCO devices. As data-plane standards did not conclude, such extensions were considered pre-mature. So by Sept. 2016 no common reference model exists as a basis that would allow to define a data model for modulated optical interfaces with coherent detection receivers.

7. Addressing the gap in controlling PRESCO-DWDM interfaces

As the Industry is surpassing standards development in providing PRESCO applications, it would benefit from commonality in implementing yang data models to control PRESCO-DWDM Modules. Existing PRESCO Modules already provide extensive FCAPS capabilities and are used to provide commercial services. Data models for such devices are in active development but suffer from commonality. On the positive side this situation allows defining PRESCO related data models based on those FCAPS functionalities.

8. Progressing PRESCO related work

PRESCO related work should be based on creating a common abstracted model based on PRESCO-FCAPS implementations, rough consensus and running code. The aim is provide the basis to enable PRESCO applications in a consistent manner by reducing implementation differences in the data structure.

As PRESCO data models would need to evolve in parallel or even precede a data-plane standard the following list of considerations should be applied going forward:

1. Control work for PRESCO SHALL have the status INFORMATIONAL or EXPERIMENTAL as it is not standards based
2. A PRESCO model SHOULD be based on control parameters available PRESCO DWDM modules

3. PRESCO modeling SHOULD aim to fit into existing data models in IETF

4. the model SHOULD allow augmentation of parameters by vendor specific extensions

5. the model SHOULD re-using existing standard parameter definitions and encoding where possible

6. Since the full set of parameters needed to characterize PRESCO modules and their encoding are undefined, application codes are not available for use in PRESCO. Therefore controlling PRESCO Modules SHALL NOT mandate the use application codes.

7. threshold levels derived from measurement values SHOULD be adjustable such that a comparable system behaviour can be achieved.

PRESCO related work evidently needs to be separate from standards related work and we need to outline what PRESCO work is not about:

- PRESCO is NOT suggesting to perform data-plane work in IETF.
- PRESCO is NOT providing data models for non-PRESCO interfaces
- PRESCO does NOT propose to utilize Data models defined for PRESCO modules to be re-used for standard models when those become available
- PRESCO does NOT require multi-vendor compatibility of PRESCO-Modules on data plane. I.e. a pair of PRESCO-DWDM interfaces having identical parameter sets describing the data-plane is not guaranteed to be compatible by ITU-T standards.

9. Contributors

10. Acknowledgements

TBD
11. IANA Considerations

This memo includes no request to IANA.

12. Security Considerations

This document discusses the need for a non-standard YANG data Model. It has no security impact on the Internet.

13. References

13.1. Normative References

[ITU.G680]

[ITU.G697]

[ITU.G698.2]

[ITU.G872]

[ITU.G874.1]


13.2. Informative References

[ccamp-charter]

[draft-2012]

[IETF-86-ccamp-minutes]


[SG15-2012]
Appendix A. Additional Stuff

Parameters that need to be encoded addressing PRESCO applications are:

- Modulation Format
- Spectral efficiency or Bit per symbols
- baud rate
- bandwidth required by the PRESCO-DWDM carrier
- Carrier central frequency (this might not follow the ITU-T grid)
- Forward Error Correction code
- Tx and Rx power
- Frequency/Wavelength
- ...

As for PRESCO applications the completeness of these Parameters to fully characterize PRESCO interfaces is not guaranteed, additional parameters will be added as needed.

Authors’ Addresses

Gert Grammel (editor)
Juniper
Oskar-Schlemmer-Str. 15
Muenchen 80807
Germany

Phone: +1 408 940 1375
Email: ggrammel@juniper.net

Dharini Hiremagalur
Juniper
1194 N Mathilda Avenue
Sunnyvale - 94089 California
USA

Email: dharinih@juniper.net
A YANG Data Model for Microwave Radio Link

draft-mwdt-ccamp-mw-yang-01

Abstract

This document defines a YANG data model in order to control and manage the radio link interfaces, and the connectivity to packet (typically Ethernet) interfaces in a microwave/millimeter wave node.

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 June 23, 2017.

Copyright Notice

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

The following terms are used in this document:

Carrier Termination (CT) is an interface for the capacity provided over the air by a single carrier. It is typically defined by its transmitting and receiving frequencies.

Radio Link Terminal (RLT) is an interface providing packet capacity and/or TDM capacity to the associated Ethernet and/or TDM interfaces in a node and used for setting up a transport service over a microwave/millimeter wave link.

The following acronyms are used in this document:
ACM  Adaptive Coding Modulation
ATPC Automatic Transmit Power Control
CM Coding Modulation
CT Carrier Termination
RLT Radio Link Terminal
RTPC Remote Transmit Power Control
XPIC Cross Polarization Interference Cancellation
2. Introduction

This document defines a YANG data model for management and control of the radio link interface(s) and the relationship to packet (typically Ethernet) and/or TDM interfaces in a microwave/millimeter wave node. The data model includes configuration and state data.

The design of the data model follows the framework for management and control of microwave and millimeter wave interface parameters defined in [mw-fmwk]. This framework identifies the need and the scope of the YANG data model, the use cases and requirements that the model needs to support. Moreover, it provides a detailed gap analysis to identify the missing parameters and functionalities of the existing and established models to support the specified use cases and requirements, and based on that recommends how the gaps should be filled with the development of the new model.

According to the conclusion of the gap analysis, the structure of the data model is based on the structure defined in [I-D.ahlberg-ccamp-microwave-radio-link] and it augments RFC 7223 to align with the same structure for management of the packet interfaces. More specifically, the model will include interface layering to manage the capacity provided by a radio link terminal for the associated Ethernet and TDM interfaces, using the principles for interface layering described in RFC 7223 as a basis.

The designed YANG data model uses the IETF: Radio Link Model [I-D.ahlberg-ccamp-microwave-radio-link] and the ONF: Microwave Modeling [ONF-model] as the basis for the definition of the detailed leaves/parameters, and proposes new ones to cover identified gaps which are analyzed in [mw-fmwk].

3. YANG Data Model (Tree Structure)
Internet-Draft    Microwave YANG Model        December 2016

+--rw rx-frequency-config?         boolean
+--rw duplex-distance              uint32
+--rw channel-separation           decimal64
+--rw polarization?                enumeration
+--rw power-mode                   enumeration
+--rw selected-output-power        power
+--rw atpc-lower-threshold         power
+--rw atpc-upper-threshold         power
+--rw coding-modulation-mode       enumeration
+--rw selected-cm                  identityref
+--rw selected-min-acm             identityref
+--rw selected-max-acm             identityref
+--rw if-loop?                     enumeration
+--rw rf-loop?                     enumeration
+--rw ct-performance-thresholds
  +--rw received-level-alarm-threshold?   power
  +--rw transmitted-level-alarm-threshold?   power
  +--rw ber-alarm-threshold?            enumeration
augment /if:interfaces-state/if:interface:
  +--ro tx-oper-status?             enumeration
  +--ro actual-transmitted-level?   power
  +--ro actual-received-level?      power
  +--ro actual-tx-cm?               identityref
  +--ro actual-snir?                decimal64
  +--ro actual-xpi?                 decimal64 {xpic}?
  +--ro capabilities
    +--ro min-tx-frequency?         uint32
    +--ro max-tx-frequency?         uint32
    +--ro min-rx-frequency?         uint32
    +--ro max-rx-frequency?         uint32
    +--ro available-min-output-power?   power
    +--ro available-max-output-power?   power
    +--ro available-min-acm?         identityref
    +--ro available-max-acm?         identityref
augment /if:interfaces-state/if:interface/if:statistics:
  +--ro bbe?        yang:counter32
  +--ro es?         yang:counter32
  +--ro ses?        yang:counter32
  +--ro uas?        yang:counter32
  +--ro min-rltm?   power
  +--ro max-rltm?   power
  +--ro min-tltm?   power
  +--ro max-tltm?   power

4. YANG Module

<CODE BEGINS> file "ietf-microwave-radio-link.yang"

module ietf-microwave-radio-link {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-microwave-radio-link";
  prefix mrl;

  import ietf-yang-types {
    prefix yang;
  }

  import ietf-interfaces {
    prefix if;
  }

  import iana-if-type {
    prefix ianaift;
  }

  organization
    "IETF CCAMP Working Group";
  contact
    "jonas.ahlberg@ericsson.com
     amy.yemin@huawei.com
     Xi.Li@neclab.eu
     cjbc@it.uc3m.es
     k-kawada@ah.jp.nec.com";

  description
    "This is a module for the entities in a generic microwave system.";

  revision 2016-12-22 {
    description
      "Draft revision covering a complete scope for configuration and state data for radio link interfaces.";
    reference "";
  }

revision 2016-10-29 {
    description
        "Draft revision.";
    reference "";
}

/*@ Features */
* Features *
*/

feature xpic {
    description
        "Indicates that the device supports XPIC.";
}

feature mimo {
    description
        "Indicates that the device supports MIMO.";
}

feature tdm {
    description
        "Indicates that the device supports TDM.";
}

/*@ Interface identities */
* Interface identities *
*/

identity radio-link-terminal {
    base ianaift:iana-interface-type;
    description
        "Interface identity for a radio link terminal.";
}

identity carrier-termination {
    base ianaift:iana-interface-type;
    description
        "Interface identity for a carrier termination.";
}

/*@ Radio-link-terminal mode identities */
identity rlt-mode {
    description
        "A description of the mode in which the radio link terminal is configured. The format is X plus Y. X represent the number of bonded carrier terminations. Y represent the number of protecting carrier terminations.";
}

identity one-plus-zero {
    base rlt-mode;
    description
        "1 carrier termination only.";
}

identity one-plus-one {
    base rlt-mode;
    description
        "1 carrier termination and 1 protecting carrier termination.";
}

identity two-plus-zero {
    base rlt-mode;
    description
        "2 bonded carrier terminations.";
}

/*
 * Coding and modulation identities
 */

identity coding-modulation {
    description
        "The coding and modulation schemes.";
}

identity half-bpsk-strong {
    base coding-modulation;
    description
        "Half BPSK strong coding and modulation scheme.";
}
identity half-bpsk {
    base coding-modulation;
    description
    "Half BPSK coding and modulation scheme.";
}

identity half-bpsk-light {
    base coding-modulation;
    description
    "Half BPSK light coding and modulation scheme.";
}

identity bpsk-strong {
    base coding-modulation;
    description
    "BPSK strong coding and modulation scheme.";
}

identity bpsk {
    base coding-modulation;
    description
    "BPSK coding and modulation scheme.";
}

identity bpsk-light {
    base coding-modulation;
    description
    "BPSK light coding and modulation scheme.";
}

identity qpsk {
    base coding-modulation;
    description
    "QPSK coding and modulation scheme.";
}

identity qam-4-strong {
    base coding-modulation;
    description
    "4 QAM strong coding and modulation scheme.";
}

identity qam-4 {
    base coding-modulation;
    description
    "4 QAM coding and modulation scheme.";
}
identity qam-4-light {
    base coding-modulation;
    description
        "4 QAM light coding and modulation scheme.";
}

identity qam-16-strong {
    base coding-modulation;
    description
        "16 QAM strong coding and modulation scheme.";
}

identity qam-16 {
    base coding-modulation;
    description
        "16 QAM coding and modulation scheme.";
}

identity qam-16-light {
    base coding-modulation;
    description
        "16 QAM light coding and modulation scheme.";
}

identity qam-32-strong {
    base coding-modulation;
    description
        "32 QAM strong coding and modulation scheme.";
}

identity qam-32 {
    base coding-modulation;
    description
        "32 QAM coding and modulation scheme.";
}

identity qam-32-light {
    base coding-modulation;
    description
        "32 QAM light coding and modulation scheme.";
}

identity qam-64-strong {
    base coding-modulation;
    description
        "64 QAM strong coding and modulation scheme.";
}
identity qam-64 {
    base coding-modulation;
    description
    "64 QAM coding and modulation scheme.";
}

identity qam-64-light {
    base coding-modulation;
    description
    "64 QAM light coding and modulation scheme.";
}

identity qam-128-strong {
    base coding-modulation;
    description
    "128 QAM strong coding and modulation scheme.";
}

identity qam-128 {
    base coding-modulation;
    description
    "128 QAM coding and modulation scheme.";
}

identity qam-128-light {
    base coding-modulation;
    description
    "128 QAM light coding and modulation scheme.";
}

identity qam-256-strong {
    base coding-modulation;
    description
    "256 QAM strong coding and modulation scheme.";
}

identity qam-256 {
    base coding-modulation;
    description
    "256 QAM coding and modulation scheme.";
}

identity qam-256-light {
    base coding-modulation;
    description
    "256 QAM light coding and modulation scheme.";
}
identity qam-512-strong {
    base coding-modulation;
    description
    "512 QAM strong coding and modulation scheme.";
}

identity qam-512 {
    base coding-modulation;
    description
    "512 QAM coding and modulation scheme.";
}

identity qam-512-light {
    base coding-modulation;
    description
    "512 QAM light coding and modulation scheme.";
}

identity qam-1024-strong {
    base coding-modulation;
    description
    "1024 QAM strong coding and modulation scheme.";
}

identity qam-1024 {
    base coding-modulation;
    description
    "1024 QAM coding and modulation scheme.";
}

identity qam-1024-light {
    base coding-modulation;
    description
    "1024 QAM light coding and modulation scheme.";
}

identity qam-2048-strong {
    base coding-modulation;
    description
    "2048 QAM strong coding and modulation scheme.";
}

identity qam-2048 {
    base coding-modulation;
    description
    "2048 QAM coding and modulation scheme.";
}
identity qam-2048-light {
  base coding-modulation;
  description
    "2048 QAM light coding and modulation scheme.";
}

identity qam-4096-strong {
  base coding-modulation;
  description
    "4096 QAM strong coding and modulation scheme.";
}

identity qam-4096 {
  base coding-modulation;
  description
    "4096 QAM coding and modulation scheme.";
}

identity qam-4096-light {
  base coding-modulation;
  description
    "4096 QAM light coding and modulation scheme.";
}

identity protection-architecture-type {
  description
    "protection architecture type";
}

identity one-plus-one-type {
  base protection-architecture-type;
  description
    "One carrier termination and
    one protecting carrier termination.";
}

identity one-to-n-type {
  base protection-architecture-type;
  description
    "One carrier termination protecting
    n other carrier terminations.";
}
identity protection-states {
    description
        "Identities describing the status of the protection, in a group of carrier terminations configured in a radio link protection mode.";
}

identity unprotected {
    base protection-states;
    description "Not protected";
}

identity protected {
    base protection-states;
    description "Protected";
}

identity unable-to-protect {
    base protection-states;
    description "Unable to protect";
}

identity protection-external-commands{
    description
        "Protection external commands for trouble shooting purpose.";
}

identity manual-switch{
    base protection-external-commands;
    description
        "A switch action initiated by an operator command. It switches normal traffic signal to the protection transport entity.";
}
identity tdm-type {
  description
      "A description of the type of TDM connection, also indicating the supported capacity of the connection.";
}

identity E1 {
  base tdm-type;
  description
      "E1 connection, 2,048 Mbit/s.";
}

identity STM-1 {
  base tdm-type;
  description
      "STM-1 connection, 155,52 Mbit/s.";
}

typedef power {
  type decimal64 {
      fraction-digits 1;
  }
  description
      "Type used for power values, selected and measured.";
}

augment "/if:interfaces/if:interface" {
  when "if:type = 'mrl:radio-link-terminal'"
  description
      "Addition of data nodes for radio link terminal to the standard Interface data model, for interfaces of the type 'radio-link-terminal'.";
leaf id {
  type string;
  default "";
  description
  "ID of the radio link terminal. Used by far-end when
  checking that it’s connected to the correct RLT."
}

leaf mode {
  type identityref {
    base rlt-mode;
  }
  mandatory true;
  description
  "A description of the mode in which the radio link
  terminal is configured. The format is X plus Y.
  X represent the number of bonded carrier terminations.
  y represent the number of protecting carrier
  terminations.";
}

leaf-list carrier-terminations {
  type if:interface-ref;
  must "/if:interfaces/if:interface[if:name = current()]"
  + "/if:type = 'mrl:carrier-termination'" {
    description
    "The type of interface must be
    'carrier-termination'.";
  }
  min-elements 1;
  description
  "A list of references to carrier terminations
  included in the radio link terminal.";
}

leaf-list rlp-groups {
  type leafref {
    path "/mrl:radio-link-protection-groups/
    + "mrl:radio-link-protection-group/mrl:name";
  }
  description
  "A list of references to the carrier termination
  groups configured for radio link protection in this
  radio link terminal.";
}

leaf-list xpic-pairs {
  if-feature xpic;
  type leafref {
    path "'/mrl:xpic-pairs/mrl:xpic-pair/mrl:name'";
  }
  description
    "A list of references to the XPIC pairs used in this radio link terminal. One pair can be used by two terminals.";
}

leaf mimo-group {
  if-feature mimo;
  type leafref {
    path "'/mrl:mimo-groups/mrl:mimo-group/mrl:name'";
  }
  description
    "A reference to the MIMO group used in this radio link terminal. One group can be used by more than one terminal.";
}

list tdm-connections {
  if-feature tdm;
  key "tdm-type";
  description
    "A list stating the number of TDM connections of a specified tdm-type that is supported by the RLT.";
  leaf tdm-type {
    type identityref {
      base tdm-type;
    }
    description
      "The type of TDM connection, which also indicates the supported capacity.";
  }
  leaf tdm-connections {
    type uint16;
    mandatory true;
    description "Number of connections of the specified type.";
  }
}
/*
 * Carrier Termination - Configuration data nodes
 */

augment "/if:interfaces/if:interface" {
    when "if:type = 'mrl:carrier-termination'";
    description "Addition of data nodes for carrier termination to
    the standard Interface data model, for interfaces
    of the type 'carrier-termination'.";
    leaf carrier-id {
        type string;
        default "A";
        description "ID of the carrier. (e.g. A, B, C or D)
        Used in XPIC & MIMO configurations to check that
        the carrier termination is connected to the correct
        far-end carrier termination. Should be the same
        carrier ID on both sides of the hop.
        Defaulted when not MIMO or XPI.");
    }
    leaf tx-enabled {
        type boolean;
        default "false";
        description " Disables (false) or enables (true) the
        transmitter.Only applicable when the interface
        is enabled (interface:enabled = true) otherwise
        it’s always disabled.";
    }
    leaf tx-frequency {
        type uint32;
        units "kHz";
        mandatory true;
        description "Selected transmitter frequency.";
    }
}

leaf rx-frequency {
    type uint32;
    units "kHz";
    description
    "Selected receiver frequency. Mandatory and writeable when rx-frequency-config=true. Otherwise read-only and calculated from tx-frequency and duplex-distance."
}

leaf rx-frequency-config {
    type boolean;
    default "true";
    description
    "Enable (true) or disable (false) direct configuration of rx-frequency and instead using a defined duplex distance."
}

leaf duplex-distance {
    when "../rx-frequency-config = 'false'";
    type uint32;
    units "kHz";
    mandatory true;
    description
    "Distance between Tx & Rx frequencies. Used to calculate rx-frequency when rx-frequency-config=false."
}

leaf channel-separation {
    type decimal64 {
        fraction-digits 1;
    }
    units "MHz";
    mandatory true;
    description
    "The amount of bandwidth allocated to a carrier."
}

leaf polarization {
    type enumeration {
        enum "horizontal" {
            description "Horizontal polarization."
        }
    }
}
enum "vertical" {
  description "Vertical polarization.";
}
enum "not-specified" {
  description "Polarization not specified.";
}
default "not-specified";
description
  "Polarization - A textual description for info only.";
}

leaf power-mode {
  type enumeration {
    enum rtpc {
      description "Remote Transmit Power Control (RTPC).";
    }
    enum atpc {
      description "Automatic Transmit Power Control (ATPC).";
    }
  }
  mandatory true;
description
  "A choice of Remote Transmit Power Control (RTPC) or Automatic Transmit Power Control (ATPC).";
}

leaf selected-output-power {
  type power {
    range "-99..40";
  }
  units "dBm";
  mandatory true;
description
  "Selected output power in RTPC mode and selected maximum selected maximum output power in ATPC mode. Minimum output power in ATPC mode is the same as the system capability, available-min-output-power.";
}

leaf atpc-lower-threshold {
  when "/..../power-mode = 'atpc'";
  type power {
    range "-99..-30";
  }
  units "dBm";
}
mandatory true;
description
  "The lower threshold for the input power at far-end used in
  the ATPC mode."
  
leaf atpc-upper-threshold {
  when "./power-mode = 'atpc'"
  type power {
    range "-99..-30"
  }
  units "dBm"
  mandatory true
  description
    "The upper threshold for the input power
    at far-end used in the ATPC mode."
  
leaf coding-modulation-mode {
  type enumeration {
    enum fixed {
      description "Fixed coding/modulation."
    }
    enum adaptive {
      description "Adaptive coding/modulation."
    }
  }
  mandatory true
  description
    "A selection of fixed or
    adaptive coding/modulation mode."
  
leaf selected-cm {
  when "./coding-modulation-mode = 'fixed'"
  type identityref {
    base coding-modulation;
  }
  mandatory true
  description
    "Selected fixed coding/modulation."
}
leaf selected-min-acm {
  when "../coding-modulation-mode = 'adaptive'";
  type identityref {
    base coding-modulation;
  }
  mandatory true;
  description
  "Selected minimum coding/modulation. Adaptive coding/modulation shall not go below this value.";
}

leaf selected-max-acm {
  when "../coding-modulation-mode = 'adaptive'";
  type identityref {
    base coding-modulation;
  }
  mandatory true;
  description
  "Selected maximum coding/modulation. Adaptive coding/modulation shall not go above this value.";
}

leaf if-loop {
  type enumeration {
    enum disabled {
      description "Disables the IF Loop.";
    }
    enum client {
      description "Loops the signal back to the client side.";
    }
    enum radio {
      description "Loops the signal back to the radio side.";
    }
  }
  default "disabled";
  description
  "Enable (client/radio) or disable (disabled) the IF loop, which loops the signal back to the client side or the radio side.";
}
leaf rf-loop {
  type enumeration {
    enum disabled {
      description "Disables the RF Loop.";
    }
    enum client {
      description "Loops the signal back to the client side.";
    }
    enum radio {
      description "Loops the signal back to the radio side.";
    }
  }
  default "disabled";
  description "Enable (client/radio) or disable (disabled) the RF loop, which loops the signal back to the client side or the radio side.";
}

container ct-performance-thresholds {
  description "Specification of thresholds for when alarms should be sent and cleared for various performance counters.";
  leaf received-level-alarm-threshold {
    type power {
      range "-99..-30";
    }
    units "dBm";
    default "-99";
    description "Specification of at which received power level an alarm should be raised.";
  }
  leaf transmitted-level-alarm-threshold {
    type power {
      range "-99..40";
    }
    units "dBm";
    default "-99";
    description "An alarm is sent when the transmitted power level is below the specified threshold.";
  }
}
leaf ber-alarm-threshold {
  type enumeration {
    enum "10e-9" {
      description "Threshold at 10e-9.";
    }
    enum "10e-8" {
      description "Threshold at 10e-8.";
    }
    enum "10e-7" {
      description "Threshold at 10e-7.";
    }
    enum "10e-6" {
      description "Threshold at 10e-6.";
    }
    enum "10e-5" {
      description "Threshold at 10e-5.";
    }
    enum "10e-4" {
      description "Threshold at 10e-4.";
    }
    enum "10e-3" {
      description "Threshold at 10e-3.";
    }
    enum "10e-2" {
      description "Threshold at 10e-2.";
    }
    enum "10e-1" {
      description "Threshold at 10e-1.";
    }
  }
  default "10e-6";
  description
    "Specification of at which BER an alarm should be raised.";
}

/*
 * Radio Link Terminal - Operational state data nodes
 * Currently nothing in addition to the general
 * interface-state model.
 */

/*
 * Carrier Termination - Operational state data nodes
 */

augment "/if:interfaces-state/if:interface" {
    when "if:type = 'mrl:carrier-termination'";
    description
        "Addition of state data nodes for carrier termination to
        the standard Interface state data model, for interfaces
        of the type 'carrier-termination'.";

    leaf tx-oper-status {
        type enumeration {
            enum "off" {
                description "Transmitter is off.";
            }
            enum "on" {
                description "Transmitter is on.";
            }
            enum "standby" {
                description "Transmitter is in standby.";
            }
        }
        description
            "Shows the operative status of the transmitter.";
    }

    leaf actual-transmitted-level {
        type power {
            range "-99..40";
        }
        units "dBm";
        description
            "Actual transmitted power level (0.1 dBm resolution).";
    }

    leaf actual-received-level {
        type power {
            range "-99..-20";
        }
        units "dBm";
        description
            "Actual received power level (0.1 dBm resolution).";
    }

    leaf actual-tx-cm {
        type identityref {
            base coding-modulation;
        }
        description
            "Actual coding/modulation in transmitting direction.";
    }
}
leaf actual-snir {
  type decimal64 {
    fraction-digits 1;
    range "0..99";
  }
  units "dB";
  description
    "Actual signal to noise plus interference ratio.
     (0.1 dB resolution).";
}

leaf actual-xpi {
  if-feature xpic;
  type decimal64 {
    fraction-digits 1;
    range "0..99";
  }
  units "dB";
  description
    "The actual carrier to cross-polar interference.
     Only valid if XPIC is enabled. (0.1 dB resolution).";
}

container capabilities {
  description
    "Capabilities of the the installed equipment and
     some selected configurations.";

  leaf min-tx-frequency {
    type uint32;
    units "kHz";
    description
      "Minimum Tx frequency possible to use.";
  }

  leaf max-tx-frequency {
    type uint32;
    units "kHz";
    description
      "Maximum Tx frequency possible to use.";
  }
leaf min-rx-frequency {
    type uint32;
    units "kHz";
    description
        "Minimum Rx frequency possible to use.";
}

leaf max-rx-frequency {
    type uint32;
    units "kHz";
    description
        "Maximum Tx frequency possible to use.";
}

leaf available-min-output-power {
    type power;
    units "dBm";
    description
        "The minimum output power supported.";
}

leaf available-max-output-power {
    type power;
    units "dBm";
    description
        "The maximum output power supported.";
}

leaf available-min-acm {
    type identityref {
        base coding-modulation;
    }
    description
        "Minimum coding-modulation possible to use.";
}

leaf available-max-acm {
    type identityref {
        base coding-modulation;
    }
    description
        "Maximum coding-modulation possible to use.";
}
}
augment "/if:interfaces-state/if:interface/if:statistics" {
  when ".../if:type = 'mrl:carrier-termination'";
  description
    "Addition of state data nodes in the container statistics
     for carrier terminations to the standard Interface data
     model, for interfaces of the type 'carrier-termination'.";

  leaf bbe {
    type yang:counter32;
    units "number of block errors";
    description
      "Number of Background Block Errors (BBE) during the
       interval. A BBE is an errored block not occurring as
       part of an SES.";
  }

  leaf es {
    type yang:counter32;
    units "seconds";
    description
      "Number of Errored Seconds (ES) since last
       reset. An ES is a one-second period with
       one or more errored blocks or at least one
       defect.";
  }

  leaf ses {
    type yang:counter32;
    units "seconds";
    description
      "Number of Severely Errored Seconds (SES) during the
       interval. SES is a one-second period which contains
       equal or more than 30% errored blocks or at least
       one defect. SES is a subset of ES.";
  }

  leaf uas {
    type yang:counter32;
    units "seconds";
    description
      "Number of Unavailable Seconds (UAS), that is, the
       total time that the node has been unavailable during
       a fixed measurement interval.";
  }
}
leaf min-rltm {
    type power {
        range "-99..-20";
    }
    units "dBm";
    description
        "Minimum received power level since last reset.";
}

leaf max-rltm {
    type power {
        range "-99..-20";
    }
    units "dBm";
    description
        "Maximum received power level since last reset.";
}

leaf min-tltm {
    type power {
        range "-99..40";
    }
    units "dBm";
    description
        "Minimum transmitted power level since last reset.";
}

leaf max-tltm {
    type power {
        range "-99..40";
    }
    units "dBm";
    description
        "Maximum transmitted power level since last reset.";
}

container radio-link-protection-groups {
    description
        "Configuration of radio link protected groups
        (1+1) of carrier terminations in a radio link.
        More than one protected group per radio-link-terminal is allowed.";
}
list radio-link-protection-group {
  key "name";
  description
  "List of protected groups of carrier terminations
  in a radio link."

  leaf name {
    type string;
    description
    "Name used for identification of the radio
     link protection group";
  }

  leaf protection-architecture-type {
    type identityref{
      base protection-architecture-type;
    }
    default "one-plus-one-type";
    description
    "The type of protection architecture
     used, e.g. one carrier termination
     protecting one carrier termination."
  }

  leaf protection-operation-type {
    type enumeration {
      enum "non-revertive" {
        description
        "In non revertive operation, the
         traffic does not return to the
         working carrier termination if the
         switch requests are terminated. ";
      }
      enum "revertive" {
        description
        "In revertive operation, the
         traffic always returns to (or
         remains on) the working carrier
         termination if the switch requests
         are terminated. ";
      }
    }
    default "non-revertive";
    description
    "The type of protection operation, i.e.
     revertive or non-revertive operation.";
  }
}

leaf-list working-entity {
  when "./protection-operation-type = 'revertive'";
  type if:interface-ref;
  must "/if:interfaces/if:interface[if:name = current()]" + "/if:type = 'mrl:carrier-termination'" {
    description
    "The type of a working-entity must be 'carrier-termination'.";
  }
  min-elements 1;
  description
  "The carrier terminations over which the traffic normally should be transported over when there is no need to use the protecting carrier termination.";
}

leaf revertive-wait-to-restore {
  when "./protection-operation-type = 'revertive'";
  type uint16;
  units "seconds";
  default "0";
  description
  "The time to wait before switching back to the working carrier termination if protection-operation-type is revertive.";
}

leaf-list radio-link-protection-members {
  type if:interface-ref;
  must "/if:interfaces/if:interface[if:name = current()]" + "/if:type = 'mrl:carrier-termination'" {
    description
    "The type of a protection member must be 'carrier-termination'.";
  }
  min-elements 2;
  description
  "Association to a group of carrier terminations configured for radio link protection and used in the radio link terminal.";
}
action protection-external-commands {
  input {
    leaf protection-external-command {
      type identityref {
        base protection-external-commands;
      }
      description
      "Execution of protection external
       commands for trouble shooting purpose."
    }
  }
}

/*
 * Radio Link Protection - Operational state data nodes
 */
container radio-link-protection-groups-state {
  config false;
  description
  "State data for radio link protected groups
   of carrier terminations in a radio link.";
  list radio-link-protection-group {
    key "name";
    description
    "List of protected groups of carrier
     terminations in a radio link.";
    leaf name {
      type string;
      description
      "Name used for identification of the
       radio link protection group.";
    }
    leaf protection-status {
      type identityref {
        base protection-states;
      }
      description
      "Status of the protection, in a group of
       carrier terminations configured in a
       radio link protection mode.";
    }
  }
}

container xpic-pairs {
    if-feature xpic;
    description
        "Configuration of carrier termination pairs
         for operation in XPIC mode.";

    list xpic-pair {
        key "name";
        description
            "List of carrier termination pairs in XPIC mode.";

        leaf name {
            type string;
            description
                "Name used for identification of the XPIC pair.";
        }

        leaf enabled {
            type boolean;
            default "false";
            description
                "Enable(true)/disable(false) XPIC";
        }

        leaf-list xpic-members {
            type if:interface-ref;
            must "/if:interfaces/if:interface[if:name = current()]" + "/if:type = 'mrl:carrier-termination'" {
                description
                    "The type of a xpic-member must be
                     'carrier-termination'.";
            }
            min-elements 2;
            max-elements 2;
            description
                "Association to XPIC pairs used in the
                 radio link terminal.";
        }
    }
}
container mimo-groups {
    if-feature mimo;
    description
        "Configuration of carrier terminations for operation in MIMO mode."

    list mimo-group {
        key "name";
        description
            "List of carrier terminations in MIMO mode."

        leaf name {
            type string;
            description
                "Name used for identification of the MIMO group."
        }

        leaf enabled {
            type boolean;
            default "false";
            description
                "Enable(true)/disable(false) MIMO"
        }

        leaf-list mimo-members {
            type if:interface-ref;
            must "/if:interfaces/if:interface[if:name = current()]" + "/if:type = 'mrl:carrier-termination'" {
                description
                    "The type of a mimo-member must be 'carrier-termination'."
            }
            min-elements 2;
            description
                "Association to a MIMO group if used in the radio link terminal."
        }
    }
}
5. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

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

The security considerations of [RFC7223] also apply to this document.

6. IANA Considerations

TBD.

7. References

7.1. Normative References


7.2. Informative References


OTN Service YANG Model
draft-sharma-ccamp-otn-service-model-00

Abstract

This document describes the YANG data model for OTN Services.

Status of this Memo

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

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

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

This Internet-Draft will expire on January 7, 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. Model Overview ............................................... 3
   2.1. OTN Mux Service ..................................... 3
   2.2. Model Tree ........................................... 4
   2.3. OTN Service YANG Model ............................. 4
   2.4. Transport Types YANG Model ......................... 9
3. Security Considerations ................................. 18
4. IANA Considerations .................................. 18
5. Acknowledgements ........................................ 18
6. Normative References ................................. 18
Authors’ Addresses .......................................... 18
1. Introduction

OTN transport networks can carry various types of client services. In many cases, the client service is an OTN service across connected domains in a multi-domain network. These OTN services can either be transported or switched in the OTN network. If an OTN service is switched then additional parameters need to be provided to create a Mux OTN service.

This document provides YANG model for creating OTN service. The model augments the TE Tunnel model, which is an abstract model to create TE Tunnels.

2. Model Overview

This section provides an overview of the OTN Service Model.

2.1. OTN Mux Service

![OTN Mux Service Diagram]

Figure 1: OTN Mux Service in a multi-domain network topology

Figure 1 shows a multi-domain OTN network with three domains. In this example, user wants to setup an end-to-end OTN service that passes through Domain-2. In order to create an OTN mux service in Domain-2, user will need to specify the exact details of the client side LO-ODU on NE2 and NE3, so that these service endpoints can be paired with the LO-ODU endpoints on NE1 and NE4, respectively.
Let’s assume that ODU4 is the client side HO-ODU on NE2 and NE3, and the client signal is ODU2. User will need to specify the OTN client signal (ODU2 in this example), the Tributary Port Number (TPN), Tributary Slot Granularities (TSG) and timeslots to be used. As shown in the figure above, these service parameters must be the same between NE1 and NE2, and NE3 and NE4.

Once the OTN Mux service is setup in Domain-2, the incoming signal from either NE1 and/or NE4 will be switched inside Domain-2, and delivered to NE at the other end.

2.2. Model Tree

module: ietf-otn-service
augment /te:te/te:tunnels/te:tunnel/te:config:
  +-rw payload-treatment?  enumeration
  +-rw src-client-signal?   identityref
  +-rw src-tpn?            uint16
  +-rw src-tsg?            identityref
  +-rw src-timeslot-count? uint16
  +-rw src-timeslots
    |  +-rw values*   uint8
  +-rw dst-client-signal?  identityref
  +-rw dst-tpn?            uint16
  +-rw dst-tsg?            identityref
  +-rw dst-timeslot-count? uint16
  +-rw dst-timeslots
    |  +-rw values*   uint8
augment /te:te/te:tunnels/te:tunnel/te:state:
  +-ro payload-treatment?  enumeration
  +-ro src-client-signal?  identityref
  +-ro src-tpn?            uint16
  +-ro src-tsg?            identityref
  +-ro src-timeslot-count? uint16
  +-ro src-timeslots
    |  +-ro values*   uint8
  +-ro dst-client-signal?  identityref
  +-ro dst-tpn?            uint16
  +-ro dst-tsg?            identityref
  +-ro dst-timeslot-count? uint16
  +-ro dst-timeslots
    |  +-ro values*   uint8

2.3. OTN Service YANG Model

<CODE BEGINS> file "ietf-otn-service@2016-06-24.yang"

module ietf-otn-service {
yang-version 1;
namespace "urn:ietf:params:xml:ns:yang:ietf-otn-service";
prefix "otn-svc";

import ietf-te { prefix "te"; }
import ietf-transport-types { prefix "tran-types"; }
import yang-ext { prefix ext; revision-date 2013-07-09; }

organization
"IETF CCAMP Working Group";

contact
"WG Web: <http://tools.ietf.org/wg/ccamp/>
WG List: <mailto:ccamp@ietf.org>
Editor: Anurag Sharma
<mailto:AnSharma@infinera.com>
Editor: Rajan Rao
<mailto:rrao@infinera.com>
Editor: Xian Zhang
<mailto:zhang.xian@huawei.com>";

description
"This module defines a model for OTN Services.";

revision "2016-06-24" {
  description "Initial revision";
  reference "TBD";
}

grouping otn-tunnel-endpoint {
  description "Parameters for OTN service.";
    leaf payload-treatment {
      type enumeration {
        enum switching;
        enum transport;
      }
      default switching;
      description
        "Treatment of the incoming payload. Payload can
         either be switched, or transported as is.";
    }

    leaf src-client-signal {
      type identityref {

leaf src-tpn {
    type uint16 {
        range "0..4095";
    }
    description
        "Tributary Port Number. Applicable in case of mux services.";
    reference
        "RFC7139: GMPLS Signaling Extensions for Control of Evolving G.709 Optical Transport Networks.";
}

leaf src-tsg {
    type identityref {
        base tran-types:tributary-slot-granularity;
    }
    description
        "Tributary slot granularity. Applicable in case of mux services.";
    reference
        "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
}

leaf src-timeslot-count {
    type uint16;
    description
        "Number of timeslots used at the source.";
}

container src-timeslots {
    description
        "A list of tributary timeslots used by the client service. Applicable in case of mux services.";
    leaf-list values {
        type uint8;
        description
            "Tributary timeslot value.";
        reference
            "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
    }
}
leaf dst-client-signal {
  type identityref {
    base tran-types:client-signal;
  }
  description
    "Client signal at the destination endpoint of the tunnel.";
}

leaf dst-tpn {
  type uint16 {
    range "0..4095";
  }
  description
    "Tributary Port Number. Applicable in case of mux services.";
  reference
    "RFC7139: GMPLS Signaling Extensions for Control of Evolving G.709 Optical Transport Networks.";
}

leaf dst-tsg {
  type identityref {
    base tran-types:tributary-slot-granularity;
  }
  description
    "Tributary slot granularity. Applicable in case of mux services.";
  reference
    "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
}

leaf dst-timeslot-count {
  type uint16;
  description
    "Number of timeslots used at the destination.";
}

container dst-timeslots {
  description
    "A list of tributary timeslots used by the client service. Applicable in case of mux services.";
}
leaf-list values {
    type uint8;
    description "Tributary timeslot value."
    reference "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)"
}

Note: Comment has been given to authors of TE Tunnel model to add tunnel-types to the model in order to identify the technology type of the service.

grouping otn-service-type {
    description "Identifies the OTN Service type."
    container otn-service {
        presence "Indicates OTN Service."
        description "Its presence identifies the OTN Service type."
    }
}

augment "/te:te/tunnels/te:tunnel/te:tunnel-types" {
    description "Introduce OTN service type for tunnel."
    ext:augment-identifier otn-service-type-augment;
    uses otn-service-type;
}

Note: Comment has been given to authors of TE Tunnel model to add list of endpoints under config to support P2MP tunnel.

augment "/te:te/tunnels/te:tunnel/te:config" {
    description "Augment with additional parameters required for OTN service."
    ext:augment-identifier otn-tunnel-endpoint-config-augment;
    uses otn-tunnel-endpoint;
}

augment "/te:te/tunnels/te:tunnel/te:state" {
    description
"Augment with additional parameters required for OTN service."

ext:augment-identifier otn-tunnel-endpoint-state-augment;
uses otn-tunnel-endpoint;

/*
Note: Comment has been given to authors of TE Tunnel model to add tunnel-lifecycle-event to the model. This notification is reported for all lifecycle changes (create, delete, and update) to the tunnel or lsp.

augment "/te:tunnel-lifecycle-event" {

description
"OTN service event";
uses otn-service-type;
uses otn-tunnel-params;
list endpoint {
  key
  "endpoint-address tp-id";
  description
  "List of Tunnel Endpoints.";
  uses te:tunnel-endpoint;
  uses otn-tunnel-params;
}
/*
}
*/

<CODE ENDS>

2.4. Transport Types YANG Model

<CODE BEGINS> file "ietf-transport-types@2016-06-24.yang"

module ietf-transport-types {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-transport-types";
  prefix "tran-types";

  organization
  "IETF CCAMP Working Group";
  contact
  "WG Web: <http://tools.ietf.org/wg/ccamp/>"
  "WG List: <mailto:ccamp@ietf.org>"
  "Editor: Anurag Sharma"
  "mailto:AnSharma@infinera.com>"

description
"This module defines transport types.";

revision "2016-06-24" {
  description "Initial revision";
  reference "TBD";
}

identity tributary-slot-granularity {
  description "Tributary slot granularity.";
  reference "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
}

identity tsg-1.25G {
  base tributary-slot-granularity;
  description "1.25G tributary slot granularity.";
}

identity tsg-2.5G {
  base tributary-slot-granularity;
  description "2.5G tributary slot granularity.";
}

identity tributary-protocol-type {
  description "Base identity for protocol framing used by tributary signals.";
}

identity prot-OTU1 {
  base tributary-protocol-type;
  description "OTU1 protocol (2.66G)";
}

identity prot-OTU1e {
  base tributary-protocol-type;
description
"OTU1e protocol (11.04G)";
}

identity prot-OTU2 {
  base tributary-protocol-type;
  description
  "OTU2 protocol (10.70G)";
}

identity prot-OTU2e {
  base tributary-protocol-type;
  description
  "OTU2e protocol (11.09G) for 10G LAN PHY";
}

identity prot-OTU2f {
  base tributary-protocol-type;
  description
  "OTU2f protocol (11.32G) for transporting a 10 fiber channel.";
}

identity prot-OTU3 {
  base tributary-protocol-type;
  description
  "OTU3 protocol (43.01G)";
}

identity prot-OTU3e {
  base tributary-protocol-type;
  description
  "OTU3e protocol (44.57G) for transporting four OTU2e signals.";
}

identity prot-OTU3e2 {
  base tributary-protocol-type;
  description
  "OTU3e2 protocol (44.58G)";
}

identity prot-OTU4 {
  base tributary-protocol-type;
  description
  "OTU4 protocol (112G) for transporting 100GE
signal.

identity prot-OTUCn {
    base tributary-protocol-type;
    description
        "OTUCn protocol (beyond 100G) for transporting more than 100G signals.";
}

identity prot-ODU0 {
    base tributary-protocol-type;
    description
        "ODU0 protocol (1.24G).";
}

identity prot-ODU1 {
    base tributary-protocol-type;
    description
        "ODU1 protocol (2.49G).";
}

identity prot-ODU1e {
    base tributary-protocol-type;
    description
        "ODU1e protocol (10.35G).";
}

identity prot-ODU2 {
    base tributary-protocol-type;
    description
        "ODU2 protocol (10.03G).";
}

identity prot-ODU2e {
    base tributary-protocol-type;
    description
        "ODU2e protocol (10.39G).";
}

identity prot-ODU3 {
    base tributary-protocol-type;
    description
        "ODU 3 protocol (40.31G).";
}

identity prot-ODU3e2 {
    base tributary-protocol-type;
description
"ODU3e2 protocol (41.78G).";
}

identity prot-ODU4 {
    base tributary-protocol-type;
    description
"ODU4 protocol (104.79G).";
}

identity prot-ODUFlex-cbr {
    base tributary-protocol-type;
    description
"ODU Flex CBR protocol for transporting constant bit rate signal.";
}

identity prot-ODUFlex-gfp {
    base tributary-protocol-type;
    description
"ODU Flex GFP protocol for transporting stream of packets using Generic Framing Procedure.";
}

identity prot-ODUCn {
    base tributary-protocol-type;
    description
"ODUCn protocol (beyond 100G).";
}

identity prot-1GbE {
    base tributary-protocol-type;
    description
"1G Ethernet protocol";
}

identity prot-10GbE-LAN {
    base tributary-protocol-type;
    description
"10G Ethernet LAN protocol";
}

identity prot-40GbE {
    base tributary-protocol-type;
    description
"40G Ethernet protocol";
}
identity prot-100GbE {
    base tributary-protocol-type;
    description
    "100G Ethernet protocol";
}

identity client-signal {
    description
    "Base identity from which specific client signals for the tunnel are derived.";
}

identity client-signal-1GbE {
    base client-signal;
    description
    "Client signal type of 1GbE";
}

identity client-signal-10GbE-LAN {
    base client-signal;
    description
    "Client signal type of 10GbE LAN";
}

identity client-signal-10GbE-WAN {
    base client-signal;
    description
    "Client signal type of 10GbE WAN";
}

identity client-signal-40GbE {
    base client-signal;
    description
    "Client signal type of 40GbE";
}

identity client-signal-100GbE {
    base client-signal;
    description
    "Client signal type of 100GbE";
}

identity client-signal-OC3_STM1 {
    base client-signal;
    description
    "Client signal type of OC3 and STM1";
}
identity client-signal-OC12_STM4 {
    base client-signal;
    description
    "Client signal type of OC12 and STM4";
}

identity client-signal-OC48_STM16 {
    base client-signal;
    description
    "Client signal type of OC48 and STM16";
}

identity client-signal-OC192_STM64 {
    base client-signal;
    description
    "Client signal type of OC192 and STM64";
}

identity client-signal-OC768_STM256 {
    base client-signal;
    description
    "Client signal type of OC768 and STM256";
}

identity client-signal-OTU1 {
    base client-signal;
    description
    "Client signal type of OTU1 (2.66G)";
}

identity client-signal-OTU2 {
    base client-signal;
    description
    "Client signal type of OTU2 (10.70G)";
}

identity client-signal-OTU2e {
    base client-signal;
    description
    "Client signal type of OTU2e (11.09G)";
}

identity client-signal-OTU2f {
    base client-signal;
    description
    "Client signal type of OTU2f (11.32G)";
}
identity client-signal-OTU3 {
    base client-signal;
    description
    "Client signal type of OTU3 (43.01G)";
}

identity client-signal-OTU3e {
    base client-signal;
    description
    "Client signal type of OTU3e (44.58G)";
}

identity client-signal-OTU4 {
    base client-signal;
    description
    "Client signal type of OTU4 (112G)";
}

identity client-signal-OTUCn {
    base client-signal;
    description
    "Client signal type of OTUCn (beyond 100G)";
}

identity client-signal-ODU0 {
    base client-signal;
    description
    "Client signal type of ODU0 (1.24G)";
}

identity client-signal-ODU1 {
    base client-signal;
    description
    "ODU1 protocol (2.49G)";
}

identity client-signal-ODU2 {
    base client-signal;
    description
    "Client signal type of ODU2 (10.03G)";
}

identity client-signal-ODU2e {
    base client-signal;
    description
    "Client signal type of ODU2e (10.39G)";
}
identity client-signal-ODU3 {
    base client-signal;
    description
        "Client signal type of ODU 3 (40.31G)";
}

identity client-signal-ODU3e2 {
    base client-signal;
    description
        "Client signal type of ODU3e2 (41.78G)";
}

identity client-signal-ODU4 {
    base client-signal;
    description
        "Client signal type of ODU4 (104.79G)";
}

identity client-signal-ODUFlex-cbr {
    base client-signal;
    description
        "Client signal type of ODU Flex CBR";
}

identity client-signal-ODUFlex-gfp {
    base client-signal;
    description
        "Client signal type of ODU Flex GFP";
}

identity client-signal-ODUCn {
    base client-signal;
    description
        "Client signal type of ODUCn (beyond 100G).";
}

identity client-signal-FC400 {
    base client-signal;
    description
        "Client signal type of Fibre Channel FC400.";
}

identity client-signal-FC800 {
    base client-signal;
    description
        "Client signal type of Fibre Channel FC800.";
}
identity client-signal-FICON-4G {
  base client-signal;
  description
    "Client signal type of Fibre Connection 4G."
}

identity client-signal-FICON-8G {
  base client-signal;
  description
    "Client signal type of Fibre Connection 8G."
}

3. Security Considerations

TBD

4. IANA Considerations

TBD

5. Acknowledgements

6. Normative References


Authors’ Addresses

Anurag Sharma
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA
Phone: +1-408-572-5365
Email: AnSharma@infinera.com

Rajan Rao
Infinera Corp
169 Java Drive
Sunnyvale, CA  94089
USA
Phone: +1-408-543-7755
Email: rrao@infinera.com

Xian Zhang
Huawei Technologies
F3-5-B R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong  518129
P.R.China
Email: zhang.xian@huawei.com
GMPLS Routing and Signalling Framework for ODUCn

draft-wang-ccamp-oducn-fwk-00

Abstract

This document provides a framework to address the GMPLS routing and
signalling issues to support Generalized Multi-Protocol Label
Switching (GMPLS) control of Optical Transport Networks (OTNs) as

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 4, 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.
1. Introduction

Currently, Optical Transport Networks (OTNs) is widely used in the transport network. Some operators already use control-plane capabilities based on GMPLS to control optical transport network to improve the network management efficiency.

The GMPLS signalling extensions defined in [RFC4328] provide the mechanisms for basic GMPLS control of OTN based on the 2001 revision of the G.709 specification. The 2012 revision of the G.709 specification, [G709-2012], introduce some new features, and the GMPLS control of OTN based on the 2012 revision of the G.709 specification is covered in [RFC7062], [RFC7096], [RFC7138] and [RFC7139]. The 2016 revision of the G.709 specification includes some new features, such as OTUCn, ODUCn and OPUCn. The OTUCn contains an optical data unit (ODUCn) and the ODUCn contains an optical payload unit (OPUCn). OTUCn, ODUCn and OPUCn are presented in an interface independent manner, by means of n OTUC, ODUC and OPUC instances that are marked #1 to #n through inverse multiplexing.

This document reviews relevant aspects of OTN technology evolution that affect the GMPLS control-plane protocols, examines why and how to update the mechanisms described in former G.709 related documents and describes the framework and solution for GMPLS control of ODUCn network.
For the purposes of the control plane, the OTN can be considered to be comprised of ODU and wavelength (Optical Channel (OCh)/ Optical Tributary Signal (OTSi)) layers. This document focuses on the control of the ODU layer, with control of the wavelength layer considered out of the scope.

1.1. Requirements Language

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

2. Terminology

OPUCn Optical Payload Unit-Cn
ODUCn Optical Data Unit-Cn
OTUCn completely standardized Optical Transport Unit-Cn
OTUCn-M Optical Transport Unit-Cn with n OxUC overhead instances and M 5G tributary slots
OTUCn completely standardized Optical Transport Unit-Cn

3. G.709 Optical Transport Network

This section provides an informative overview of the aspects of the OTN impacting control-plane protocols. This overview is based on the ITU-T Recommendations that contain the normative definition of the OTN. Technical details regarding OTN architecture and interfaces are provided in the relevant ITU-T Recommendations.

3.1. OTN ODUCn layer network

Figure 1 shows a simplified signal hierarchy of OTN ODUCn, which illustrates the layers that are related to control plane.

```
client signal (OTN clients)
    |          ODUCn
    |          OTUCn
```

Figure 1: OTN ODUCn Signal Hierarchy

ODUCn can no be used to support non-OTN client signal. OTN client signals (e.g. ODU0, ODU1, ODU2, ODU2e, ODU3, ODU4, ODUflex) are
mapped into an ODUCn container, ODUCn container is then multiplexed into OTUCn. The approximate bit rates of these signals are defined in [G709-2016] and are reproduced in Figure 2.

<table>
<thead>
<tr>
<th>ODU Type</th>
<th>ODU nominal bit rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODU0</td>
<td>1,244,160 Kbps</td>
</tr>
<tr>
<td>ODU1</td>
<td>239/238 x 2,488,320 Kbps</td>
</tr>
<tr>
<td>ODU2</td>
<td>239/237 x 9,953,280 Kbps</td>
</tr>
<tr>
<td>ODU3</td>
<td>239/236 x 39,813,120 Kbps</td>
</tr>
<tr>
<td>ODU4</td>
<td>239/227 x 99,532,800 Kbps</td>
</tr>
<tr>
<td>ODUCn</td>
<td>n x 239/226 x 99,532 800 kbit/s</td>
</tr>
<tr>
<td>ODU2e</td>
<td>239/237 x 10,312,500 Kbps</td>
</tr>
<tr>
<td>ODUflex for Constant Bit Rate Client signals</td>
<td>239/238 x client signal bit rate</td>
</tr>
<tr>
<td>ODUflex for Generic Framing Procedure - Framed (GFP-F) Mapped client signal</td>
<td>Configured bit rate</td>
</tr>
<tr>
<td>ODUflex for IMP mapped client signals</td>
<td>s x 239/238 x 5 156 250 kbit/s</td>
</tr>
<tr>
<td>s = 2, 8, n x 5 with n &gt;= 1</td>
<td></td>
</tr>
<tr>
<td>ODUflex for FlexE aware client signals</td>
<td>103 125 000 x 240/238 x n/20 kbit/s</td>
</tr>
<tr>
<td>(n = n1 + n2 + .. + np)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: ODU Types and Bit Rates

3.2. Time Slot Granularity

The initial versions of G.709 referenced by [RFC4328] only provided a single TS granularity, nominally 2.5 Gbps. [G709-2012] added an additional TS granularity, nominally 1.25 Gbps. [G709-2012] added another 5 Gbps TS granularity specially for ODUCn. The number of tributary slots (TS) defined in [G709-2016] for each ODU are reproduced in Figure 3.
<table>
<thead>
<tr>
<th>ODU Server</th>
<th>Nominal TS capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.25 Gbit/s</td>
</tr>
<tr>
<td>ODU0</td>
<td>1</td>
</tr>
<tr>
<td>ODU1</td>
<td>2</td>
</tr>
<tr>
<td>ODU2</td>
<td>8</td>
</tr>
<tr>
<td>ODU3</td>
<td>32</td>
</tr>
<tr>
<td>ODU4</td>
<td>80</td>
</tr>
<tr>
<td>ODUCn</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Figure 3: Number of tributary slots (TS)

3.3. Structure of MSI Information

When multiplexing an OTN client signal into ODUCn, [G.709-2016] specifies the information that has to be transported in-band in order to allow for correct demultiplexing. This information, known as MSI, is transported in the OPUCn overhead and is local to each link.

The MSI information is organized as a set of entries, with n entries for each OPUC TS. The MSI indicates the ODTU content of each tributary slot of an OPU. Two bytes are used for each tributary slot. The information carried by each entry is:

- TS availability bit 1 indicates if the tributary slot is available or unavailable.
- The TS occupation bit 9 indicates if the tributary slot is allocated or unallocated.
- Payload Type: the type of the transported payload.
- TPN: the port number of the OTN client signal transported by the ODUCn. The TPN is the same for all the TSs assigned to the transport of the same OTN client signal.
3.4. OTUCn sub rates (OTUCn-M)

An OTUCn with a bit rate that is not an integer multiple of 100 Gbit/s is described as an OTUCn M, it carries n instances of OTUc overhead, ODUC overhead and OPUC overhead together with M 5Gbit/s OPUCn TS. An ODUCn M and OPUCn M are not defined. When an OTUCn M is used to carry an ODUCn (20n-M) TS are marked as unavailable, in the OPUCn multiplex structure identifier (MSI), since they cannot be used to carry a client.

4. Connection Management of ODUCn

ODUCn based connection management is concerned with controlling the connectivity of ODUCn paths. As described in [G.872], The ODUk subnetwork does not support an ODUCn, which means intermediate ODUCn points do not support the switching of ODUCn time slot, intermediate ODUCn point only functions as a forwarding point. Once an ODUCn path is used to transport client signal, the TS occupied will not changed across the ODUCn network.

5. GMPLS Implications

The purpose of this section is to provide a set of requirements to be evaluated for extensions of the current GMPLS protocol suite to encompass OTN enhancements and connection management.

5.1. Implications for GMPLS Signalling

As described in Section 3, [G709-2016] introduced some new features, such as OTUCn, ODUCn and OPUCn. The mechanisms defined in [RFC4328] and [RFC7139] do not support such new OTN features, and protocol extensions will be necessary to allow them to be controlled by a GMPLS control plane. The following signalling aspects should be considered:

- Support for specifying new signal types and related traffic information. The traffic parameters should be extended in a signalling message to support the new ODUCn
- Support for LSP setup using different TS granularity
- Support for LSP setup of new ODUCn containers with related mapping and multiplexing capabilities
- Support for TPN allocation and negotiation
- Support for LSP setup of OTUCn sub rates (OTUCn-M) path
Note: ODU Virtual Concatenation (VCAT) and Link Capacity Adjustment Scheme (LCAS) is not supported in ODUCn network.

5.2. Implications for GMPLS Routing

The path computation process needs to select a suitable route for an ODUCn connection request. In order to perform the path computation, it needs to evaluate the available bandwidth on one or more candidate links. The routing protocol should be extended to convey sufficient information to represent ODU Traffic Engineering (TE) topology. Following requirements should be considered:

- Support for Tributary Slot Granularity advertisement
- Support for carrying the link multiplexing capability

The routing protocol should be able to indicate which link supports the ODUCn forwarding.

- Support for advertisement of OTUCn sub rates support information

5.3. Implications for Control-Plane Backward Compatibility

TBD

6. Solutions

TBD

7. Security Considerations

TBD

8. IANA Considerations

TBD

9. References

9.1. Normative References


9.2. Informative References

Authors' Addresses

Qilei Wang (editor)
ZTE
Nanjing
CN

Email: wang.qilei@zte.com.cn

Yuanbin Zhang
ZTE
Beijing
CN

Email: zhang.yuanbin@zte.com.cn
A YANG Data Model for Optical Transport Network Topology
draft-zhang-ccamp-l1-topo-yang-05

Abstract

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic transparently across the server-layer network resources. A transport network can be constructed from equipments utilizing any of a number of different transport technologies such as the evolving Optical Transport Networks (OTN) or packet transport as provided by the MPLS-Transport Profile (MPLS-TP).

This draft describes a YANG data model to describe the topologies of an Optical Transport Network (OTN). It is independent of control plane protocols and captures topological and resource related information pertaining to OTN. This model enables clients, which interact with a transport domain controller via a REST interface, for OTN topology related operations such as obtaining the relevant topology resource information.

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."
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 29, 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 . . . . . . . . . . . . . . . . . . . . . . . .   2
2.  Terminology and Notations . . . . . . . . . . . . . . . . .   3
3.  YANG Data Model for OTN Topology . . . . . . . . . . . . . .   4
   3.1.  the YANG Tree . . . . . . . . . . . . . . . . . . . . .   4
   3.2.  Explanation of the OTN Topology Data Model . . . . . .   5
   3.3.  The YANG Code . . . . . . . . . . . . . . . . . . . . .   5
4.  IANA Considerations . . . . . . . . . . . . . . . . . . . . .  13
5.  Manageability Considerations . . . . . . . . . . . . . . . .  13
6.  Security Considerations . . . . . . . . . . . . . . . . . . .  13
7.  Acknowledgements . . . . . . . . . . . . . . . . . . . . . .  13
8.  Contributors . . . . . . . . . . . . . . . . . . . . . . . . .  13
9.  References . . . . . . . . . . . . . . . . . . . . . . . . . .  14
   9.1.  Normative References . . . . . . . . . . . . . . . . . .  14
   9.2.  Informative References . . . . . . . . . . . . . . . . .  15
Authors’ Addresses . . . . . . . . . . . . . . . . . . . . . . .  15

1.  Introduction

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic transparently across the server-layer network resources. A transport network can be constructed of equipments utilizing any of a number of different transport technologies such as the Optical Transport Networks (OTN) or packet transport as provided by the MPLS-Transport Profile (MPLS-TP).
This document defines a data model of an OTN network topology, using YANG [RFC6020]. The model can be used by an application exposing to a transport controller via a REST interface. Furthermore, it can be used by an application for the following purposes (but not limited to):

- To obtain a whole view of the network topology information of its interest;
- To receive notifications with regard to the information change of the OTN topology;
- To enforce the establishment and update of a network topology with the characteristic specified in the data model, e.g., by a client controller;

The YANG model defined in this draft is independent of control plane protocols and captures topology related information pertaining to an Optical Transport Networks (OTN)-electrical layer, as the scope specified by [RFC7062] and [RFC7138]. Furthermore, it is not a stand-alone model, but augmenting from the TE topology YANG model defined in [I-D.ietf-teas-yang-te-topo].

Optical network technologies, including fixed Dense Wavelength Switched Optical Network (WSON) and flexible optical networks (a.k.a., flexi-grid networks), are covered in [I-D.ietf-ccamp-wson-yang] and [I-D.vergara-ccamp-flexigrid-yang], respectively.

2. Terminology and Notations

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in the YANG data tree presented later in this draft is defined in [I-D.ietf-netmod-rfc6087bis]. They are provided below for reference.

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
3. YANG Data Model for OTN Topology

3.1. the YANG Tree

```
module ietf-otn-topology
augment /nd:nets/nd:net/nd:net/nd:net-types/tet:te-topology:
  +-rw otn-topology!
augment /nd:nets/nd:net/nd:node:
  +-rw name?  string
    +-rw available-odu-info* [priority]
      |  +-rw priority  uint8
      |  +-rw odulist* [odu-type]
      |    +-rw odu-type  identityref
      |    +-rw number?  uint16
      +-rw distance?  uint32
    +-ro available-odu-info* [priority]
      |  +-ro priority  uint8
      |  +-ro odulist* [odu-type]
      |    +-ro odu-type  identityref
      |    +-ro number?  uint16
      +-ro distance?  uint32
    +-rw client-facing?  empty
    +-rw tpn?  uint16
    +-rw tag?  identityref
    +-rw protocol-type?  identityref
    +-rw adaptation-type?  adaptation-type
    +-rw sink-adapt-active?  boolean
    +-rw source-adapt-active?  boolean
    +-rw tributaryslots
      |  +-rw values*  uint8
    +-rw supported-payload-types* [index]
      |  +-rw index  uint16
      +-rw payload-type?  string
    +-ro client-facing?  empty
    +-ro tpn?  uint16
```
3.2. Explanation of the OTN Topology Data Model

As can be seen, from the data tree shown in Section 3.1, the YANG module presented in this draft augments from a more generic Traffic Engineered (TE) network topology data model, i.e., the ietf-te-topology.yang as specified in [I-D.ietf-teas-yang-te-topo]. The entities and their attributes, such as node, termination points and links, are still applicable for describing an OTN topology and the model presented in this draft only specifies with technology-specific attributes/information. For example, if the data plane complies with ITU-T G.709 (2012) standards, the switching-capability and encoding attributes MUST be filled as OTN-TDM and G.709 ODUk(Digital Path) respectively.

Note the model in this draft re-uses some attributes defined in ietf-transport-types.yang, which is specified in [I-D.sharma-ccamp-otn-service-model].

One of the main augmentations in this model is that it allows to specify the type of ODU container and the number a link can support per priority level. For example, for a ODU3 link, it may advertise 32*ODU0, 16*ODU1, 4*ODU2 available, assuming only a single priority level is supported. If one of ODU2 resource is taken to establish a ODU path, then the availability of this ODU link is updated as 24*ODU0, 12*ODU1, 3*ODU2 available. If there are equipment hardware limitations, then a subset of potential ODU type SHALL be advertised. For instance, an ODU3 link may only support 4*ODU2.

3.3. The YANG Code

<CODE BEGINS> file "ietf-otn-topology@2016-10-26.yang"

module ietf-otn-topology {
yang-version 1;
prefix "otntopo";

import ietf-network {  
  prefix "nd";
}

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

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

import ietf-transport-types {  
  prefix "tran-types";
}

organization  
  "Internet Engineering Task Force (IETF) CCAMP WG";
contact  
  "WG List: <mailto:ccamp@ietf.org>"

ID-draft editor:  
  Xian ZHANG (zhang.xian@huawei.com);  
  Anurag Sharma (AnSharma@infinera.com);

description  
  "This module defines a protocol independent Layer 1/ODU  
  topology data model.";
revision 2016-10-26 {  
  description  
    "Initial version.";
  reference  
    "draft-zhang-ccamp-l1-topo-yang-04.txt";
}

typedef  

typedef adaptation-type {
type enumeration {
    enum CBR {
        description "Constant Bit Rate.";
    }
    enum ATMvp {
        description "ATM VP."
    }
    enum GFP {
        description "Generic Framing Procedure.";
    }
    enum NULL {
        description "NULL";
    }
    enum PRBS {
        description "Pseudo Random Binary Sequence";
    }
    enum RSn {
        description "SDH/SONET section";
    }
    enum ODUj-21 {
        description "ODU payload type 21";
    }
    enum ETHERNET_PP-OS {
        description "ETHERNET_PP-OS, for ODU 2 only";
    }
    enum CBRx {
        description "CBRx(0..1.25G), for ODU0 only";
    }
    enum ODU {
        description "Optical Data Unit";
    }
}

description
"Defines a type representing the adaptation type on the termination point.";
}

/*
Groupings */

grouping otn-topology-type {
    container otn-topology {
        presence "indicates a topology type of Optical Transport Network (OTN)-electrical layer.";
        description "otn topology type";
    }
}

description "otn-topology-type";
}

grouping otn-topology-attributes {
  leaf name {
    type string;
    description "the topology name";
  }
  description "name attribute for otn topology";
}

grouping otn-node-attributes {
  description "otn-node-attributes";
  leaf name {
    type string;
    description "a name for this node.";
  }
}

grouping otn-link-attributes {
  description "otn link attributes";

  list available-odu-info{
    key "priority";
    max-elements "8";
    description "List of ODU type and number on this link";
    leaf priority {
      type uint8 {
        range "0..7";
      }
      description "priority";
    }
  }

  list odulist {
    key "odu-type";
    description "the list of available ODUs per priority level";
    leaf odu-type {
      type identityref{
        base tran-types:tributary-protocol-type;
      }
      description "the type of ODU";
    }
  }

leaf number {
    type uint16;
    description "the number of odu type supported";
}

leaf distance {
    type uint32;
    description "distance in the unit of kilometers";
}

grouping otn-tp-attributes {
    description "otn-tp-attributes";

    leaf client-facing {
        type empty;
        description
            "if present, it means this tp is a client-facing tp. adding/dropping client signal flow.";
    }

    leaf tpn {
        type uint16 {
            range "0..4095";
        }
        description
            "Tributary Port Number. Applicable in case of mux services.";
        reference
            "RFC7139: GMPLS Signaling Extensions for Control of Evolving
             G.709 Optical Transport Networks.";
    }

    leaf tsg {
        type identityref {
            base tran-types:tributary-slot-granularity;
        }
        description "Tributary slot granularity.";
        reference
            "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
    }

    leaf protocol-type {
        type identityref {
            base tran-types:tributary-protocol-type;
        }
    }
}
leaf adaptation-type {
  type adaptation-type;
  description "This attribute indicates the type of the supported adaptation function at the termination point.";
}

leaf sink-adapt-active {
  type boolean;
  description "This attribute allows for activation or deactivation of the sink adaptation function. The value of TRUE means active.";
}

leaf source-adapt-active {
  type boolean;
  description "This attribute allows for activation or deactivation of the sink adaptation function. The value of TRUE means active.";
}

container tributaryslots {
  description "A list of tributary slots used by the ODU Termination Point.";
  leaf-list values {
    type uint8;
    description "Tributary slot value.";
    reference "G.709/Y.1331, February 2012: Interfaces for the
list supported-payload-types{
  key "index";
  description "supported payload types of a TP";
  leaf index {
    type uint16;
    description "payload type index";
  }
  leaf payload-type {
    type string;
    description "the payload type supported by this client tp";
    reference "http://www.iana.org/assignments/gmpls-sig-parameters/gmpls-sig-parameters.xhtml"
    not: the payload type is defined as the generalized PIDs in GMPLS.";
  }
}

/*
* Data nodes
*/
  uses otn-topology-type;
  description "augment network types to include otn newtork";
}
augment "/nd:networks/nd:network" {
  when "/nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network";
  }
  uses otn-topology-attributes;
  description "Augment network configuration";
}
augment "/nd:networks/nd:network/nd:node" {
  when "/../nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network";
  }
}
description "Augment node configuration";
uses otn-node-attributes;
}

  when "../../../nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network."
  }
  description "Augment link configuration";
  uses otn-link-attributes;
}

  when "../../../nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network."
  }
  description "Augment link state";
  uses otn-link-attributes;
}

  when "../../../.../nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network."
  }
  description "OTN TP attributes config in a ODU topology.";
  uses otn-tp-attributes;
}

  when "../../../.../nd:network-types/tet:te-topology/otn-topology" {
    description "Augment only for otn network."
  }
  description "OTN TP attributes state in a ODU topology.";
  uses otn-tp-attributes;
}
4. IANA Considerations
   TBD.

5. Manageability Considerations
   TBD.

6. Security Considerations

   The data following the model defined in this draft is exchanged via, for example, the interface between an orchestrator and a transport network controller. The security concerns mentioned in [I-D.ietf-teas-yang-te-topo] for using ietf-te-topology.yang model also applies to this draft.

   The YANG module defined in this document can be accessed via the RESTCONF protocol defined in [I-D.ietf-netconf-restconf], or maybe via the NETCONF protocol [RFC6241].

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

   Editors note: to list specific subtrees and data nodes and their sensitivity/vulnerability.

7. Acknowledgements

   We would like to thank Igor Bryskin, Zhe Liu, Dieter Beller and Daniele Ceccarelli for their comments and discussions.

8. Contributors

   Baoquan Rao
   Huawei Technologies
   Email: raobaoquan@huawei.com

   Sergio Belotti
   Alcatel Lucent
   Email: Sergio.belotti@alcatel-lucent.com

   Huub van Helvoort
   Hai Gaoming BV
   the Netherlands
9. References

9.1. Normative References

[I-D.ietf-netconf-restconf]

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-09 (work in progress), October 2016.

[I-D.ietf-teas-yang-te-topo]

[I-D.sharma-ccamp-otn-service-model]
ansharma@infinera.com, a., Rao, R., and X. Zhang, "OTN Service YANG Model", draft-sharma-ccamp-otn-service-model-00 (work in progress), July 2016.


9.2. Informative References

[I-D.ietf-ccamp-wson-yang]

[I-D.vergara-ccamp-flexigrid-yang]

Authors' Addresses

Xian Zhang
Huawei Technologies
F3-5-B R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong  518129
P.R.China

Email: zhang.xian@huawei.com

Anurag Sharma
Infinera

Email: ansharma@infinera.com

Xufeng Liu
Ericsson

Email: xufeng.liu@ericsson.com
Abstract

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic transparently across the server-layer network resources. A transport network can be constructed from equipments utilizing any of a number of different transport technologies such as the evolving Optical Transport Networks (OTN) or packet transport as provided by the MPLS-Transport Profile (MPLS-TP).

This draft describes a YANG data model to describe the topologies of an Optical Transport Network (OTN). It is independent of control plane protocols and captures topological and resource related information pertaining to OTN. This model enables clients, which interact with a transport domain controller via a REST interface, for OTN topology related operations such as obtaining the relevant topology resource information.

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/.
1. Introduction

A transport network is a server-layer network designed to provide connectivity services for a client-layer network to carry the client traffic transparently across the server-layer network resources. A transport network can be constructed of equipments utilizing any of a number of different transport technologies such as the Optical...
Transport Networks (OTN) or packet transport as provided by the MPLS-Transport Profile (MPLS-TP).

This document defines a data model of an OTN network topology, using YANG [RFC6020]. The model can be used by an application exposing to a transport controller via a REST interface. Furthermore, it can be used by an application for the following purposes (but not limited to):

- To obtain a whole view of the network topology information of its interest;
- To receive notifications with regard to the information change of the OTN topology;
- To enforce the establishment and update of a network topology with the characteristic specified in the data model, e.g., by a client controller;

The YANG model defined in this draft is independent of control plane protocols and captures topology related information pertaining to an Optical Transport Networks (OTN)-electrical layer, as the scope specified by [RFC7062] and [RFC7138]. Furthermore, it is not a stand-alone model, but augmenting from the TE topology YANG model defined in [I-D.ietf-teas-yang-te-topo].

Optical network technologies, including fixed Dense Wavelength Switched Optical Network (WSON) and flexible optical networks (a.k.a., flexi-grid networks), are covered in [I-D.ietf-ccamp-wson-yang] and [I-D.vergara-ccamp-flexigrid-yang], respectively.

2. Terminology and Notations

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in the YANG data tree presented later in this draft is defined in [I-D.ietf-netmod-rfc6087bis]. They are provided below for reference.

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
3. YANG Data Model for OTN Topology

3.1. the YANG Tree

module: ietf-otn-topology
augment /nd:networks/nd:network/nd:network-types/tet:te-topology:
    +--rw otn-topology!
augment /nd:networks/nd:network:
    +--rw name?    string
augment /nd:networks/nd:network/nd:node:
    +--rw name?    string
augment /nd:networks/nd:network/lnk:link/tet:te/tet:config:
    +--rw available-odu-info* [priority]
        |  +--rw priority    uint8
        |  +--rw odulist* [odu-type]
        |      +--rw odu-type    identityref
        |      +--rw number?     uint16
        +--rw distance?             uint32
    +--ro available-odu-info* [priority]
        |  +--ro priority    uint8
        |  +--ro odulist* [odu-type]
        |      +--ro odu-type    identityref
        |      +--ro number?     uint16
        +--ro distance?             uint32
    +--rw client-facing?         empty
    +--rw tpn?                   uint16
    +--rw tag?                   identityref
    +--rw protocol-type?         identityref
    +--rw adaptation-type?       adaptation-type
    +--rw sink-adapt-active?     boolean
    +--rw source-adapt-active?   boolean
    +--rw tributary-slots
    |  +--rw values*   uint8
    |  +--rw supported-payload-types* [index]
    |      +--rw index   uint16
    |      +--rw payload-type? string
3.2. Explanation of the OTN Topology Data Model

As can be seen, from the data tree shown in Section 3.1, the YANG module presented in this draft augments from a more generic Traffic Engineered (TE) network topology data model, i.e., the ietf-te-topology.yang as specified in [I-D.ietf-teas-yang-te-topo]. The entities and their attributes, such as node, termination points and links, are still applicable for describing an OTN topology and the model presented in this draft only specifies with technology-specific attributes/information. For example, if the data plane complies with ITU-T G.709 (2012) standards, the switching-capability and encoding attributes MUST be filled as OTN-TDM and G.709 ODuk(Digital Path) respectively.

Note the model in this draft re-uses some attributes defined in ietf-transport-types.yang, which is specified in [I-D.sharma-ccamp-otn-tunnel-model].

One of the main augmentations in this model is that it allows to specify the type of ODU container and the number a link can support per priority level. For example, for a ODU3 link, it may advertise 32*ODU0, 16*ODU1, 4*ODU2 available, assuming only a single priority level is supported. If one of ODU2 resource is taken to establish a ODU path, then the availability of this ODU link is updated as 24*ODU0, 12*ODU1, 3*ODU2 available. If there are equipment hardware limitations, then a subset of potential ODU type SHALL be advertised. For instance, an ODU3 link may only support 4*ODU2.

3.3. The YANG Code

<CODE BEGINS> file "ietf-otn-topology@2017-04-25.yang"

module ietf-otn-topology {

---ro client-facing?   empty
---ro tpn?             uint16
---ro tsg?             identityref
---ro protocol-type?   identityref
---ro adaptation-type? adaptation-type
---ro sink-adapt-active? boolean
---ro source-adapt-active? boolean
---ro tributary-slots
  | ---ro values*   uint8
  | ---ro supported-payload-types* [index]
  |     ---ro index   uint16
  |     ---ro payload-type? string

---ro sink-adapt-active? boolean
---ro source-adapt-active? boolean
---ro tributary-slots
  | ---ro values*   uint8
  | ---ro supported-payload-types* [index]
  |     ---ro index   uint16
  |     ---ro payload-type? string
yang-version 1.1;
prefix "otntopo";

import ietf-network {
    prefix "nd";
}

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

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

import ietf-transport-types {
    prefix "tran-types";
}

organization
    "Internet Engineering Task Force (IETF) CCAMP WG";
contact
    WG List: <mailto:ccamp@ietf.org>
    ID-draft editor:
    Haomian Zheng (zhenghaomian@huawei.com);
    Zheyu Fan (fanzheyu2@huawei.com);
    Anurag Sharma (ansha@google.com);
    Xufeng Liu (Xufeng_Liu@jabil.com)
    
    description
    "This module defines a protocol independent Layer 1/ODU
topology data model."
    
    revision 2017-04-25 {
        description
        "Revision 0.3";
        reference
        "draft-zhang-ccamp-l1-topo-yang-07.txt";
    }

    /*
    typedef
    */
typedef adaptation-type {
  type enumeration {
    enum CBR {
      description "Constant Bit Rate.";
    }
    enum ATMvp {
      description "ATM VP.";
    }
    enum GFP {
      description "Generic Framing Procedure.";
    }
    enum NULL {
      description "NULL";
    }
    enum PRBS {
      description "Pseudo Random Binary Sequence";
    }
    enum RSn {
      description "SDH/SONET section";
    }
    enum ODUj-21 {
      description "ODU payload type 21";
    }
    enum ETHERNET_PP-OS {
      description "ETHERNET_PP-OS, for ODU 2 only";
    }
    enum CBRx {
      description "CBRx(0.. 1.25G), for ODU0 only";
    }
    enum ODU {
      description "Optical Data Unit";
    }
  }
}

description
"Defines a type representing the adaptation type on the termination point.";
}

/*
Groupings
*/
grouping otn-topology-type {
  container otn-topology {
    presence "indicates a topology type of Optical Transport Network (OTN)-electrical layer.";
  }
}
grouping otn-topology-attributes {
    leaf name {
        type string;
        description "the topology name";
    }
    description "name attribute for otn topology";
}

grouping otn-node-attributes {
    description "otn-node-attributes";
    leaf name {
        type string;
        description "a name for this node.";
    }
}

grouping otn-link-attributes {
    description "otn link attributes";
    list available-odu-info{
        key "priority";
        max-elements "8";
        description "List of ODU type and number on this link";
        leaf priority {
            type uint8 {
                range "0..7";
            }
            description "priority";
        }
        list odulist {
            key "odu-type";
            description "the list of available ODUs per priority level";
            leaf odu-type {
                type identityref{
                    base tran-types:tributary-protocol-type;
                }
                description "the type of ODU";
            }
        }
    }
}
leaf number {
  type uint16;
  description "the number of odu type supported";
}

leaf distance {
  type uint32;
  description "distance in the unit of kilometers";
}

grouping otn-tp-attributes {
  description "otn-tp-attributes";

  leaf client-facing {
    type empty;
    description "if present, it means this tp is a client-facing tp.
     adding/dropping client signal flow.";
  }

  leaf tpn {
    type uint16 {
      range "0..4095";
    }
    description "Tributary Port Number. Applicable in case of mux services.";
    reference "RFC7139: GMPLS Signaling Extensions for Control of Evolving G.709 Optical Transport Networks.";
  }

  leaf tsg {
    type identityref {
      base tran-types:tributary-slot-granularity;
    }
    description "Tributary slot granularity.";
    reference "G.709/Y.1331, February 2012: Interfaces for the Optical Transport Network (OTN)";
  }

  leaf protocol-type {
    type identityref {

base tran-types:tributary-protocol-type;
}
description "Protocol type for the Termination Point."
;
}
leaf adaptation-type {
  type adaptation-type;
  description
  "This attribute indicates the type of the supported
  adaptation function at the termination point.";
  reference
  "G.874.1, January 2002: Optical transport network (OTN): Protocol-neutral management information model for the
  network element view.";
}
leaf sink-adapt-active {
  type boolean;
  description
  "This attribute allows for activation or deactivation of
  the sink adaptation function. The value of TRUE means active.";
  reference
  "G.874.1, January 2002: Optical transport network (OTN): Protocol-neutral management information model for the
  network element view ";
}
leaf source-adapt-active {
  type boolean;
  description
  "This attribute allows for activation or deactivation of
  the sink adaptation function. The value of TRUE means active.";
  reference
  "G.874.1, January 2002: Optical transport network (OTN): Protocol-neutral management information model for the
  network element view ";
}
container tributary-slots {
  description
  "A list of tributary slots used by the ODU
  Termination Point.";
  leaf-list values {
    type uint8;
    description
    "Tributary slot value.";
  }
}
list supported-payload-types{
    key "index";
    description "supported payload types of a TP";
    leaf index {
        type uint16;
        description "payload type index";
    }
    leaf payload-type {
        type string;
        description "the payload type supported by this client tp";
        reference
            "http://www.iana.org/assignments/gmpls-sig-parameters
            /gmpls-sig-parameters.xhtml
            not: the payload type is defined as the generalized PIDs
            in GMPLS.";
    }
}

    uses otn-topology-type;
    description "augment network types to include otn network";
}

augment "/nd:networks/nd:network" {
    when "nd:network-types/tet:te-topology/otn-topology" {
        description "Augment only for otn network";
    }
    uses otn-topology-attributes;
    description "Augment network configuration";
}

augment "/nd:networks/nd:network/nd:node" {
    when ".../nd:network-types/tet:te-topology/otn-topology" {

    }
description "Augment only for otn network";
}
description "Augment node configuration";
uses otn-node-attributes;
}

    when "./././.nd:net-types/tet:te-topology/otn-topology" {
        description "Augment only for otn network.";
    }
    description "Augment link configuration";

    uses otn-link-attributes;
}

augment "/nd:nets/nd:net/lnk:link/tet:te/tet:state" {
    when "./././.nd:net-types/tet:te-topology/otn-topology" {
        description "Augment only for otn network.";
    }
    description "Augment link state";

    uses otn-link-attributes;
}

    when "./././.nd:net-types/tet:te-topology/otn-topology" {
        description "Augment only for otn network";
    }
    description "OTN TP attributes config in a ODU topology.";

    uses otn-tp-attributes;
}

    when "./././.nd:net-types/tet:te-topology/otn-topology" {
        description "Augment only for otn network";
    }
    description "OTN TP attributes state in a ODU topology.";

    uses otn-tp-attributes;
}

<CODE ENDS>
4. IANA Considerations

   TBD.

5. Manageability Considerations

   TBD.

6. Security Considerations

   The data following the model defined in this draft is exchanged via, for example, the interface between an orchestrator and a transport network controller. The security concerns mentioned in [I-D.ietf-teas-yang-te-topo] for using ietf-te-topology.yang model also applies to this draft.

   The YANG module defined in this document can be accessed via the RESTCONF protocol defined in [RFC8040], or maybe via the NETCONF protocol [RFC6241].

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

   Editors note: to list specific subtrees and data nodes and their sensitivity/vulnerability.

7. Acknowledgements

   We would like to thank Igor Bryskin, Zhe Liu, Dieter Beller and Daniele Ceccarelli for their comments and discussions.

8. Contributors

   Baoquan Rao  
   Huawei Technologies  
   Email: raobaoquan@huawei.com

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

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

9.1. Normative References

[I-D.ietf-teas-yang-te-topo]

[I-D.sharma-ccamp-otn-tunnel-model]
Zhang, X., xiangkun, x., ansharma@infinera.com, a., and R. Rao, "OTN Tunnel YANG Model", draft-sharma-ccamp-otn-tunnel-model-01 (work in progress), March 2017.


9.2. Informative References

[I-D.ietf-ccamp-wson-yang]

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-12 (work in progress), March 2017.

[I-D.vergara-ccamp-flexigrid-yang]


Authors’ Addresses

Haomian Zheng
Huawei Technologies
F3 R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong 518129
P.R.China

Email: zhenghaomian@huawei.com

Zheyu Fan
Huawei Technologies
F3 R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong 518129
P.R.China

Email: fanzheyu2@huawei.com

Abstract

A transport network is a lower-layer network designed to provide connectivity services for a higher-layer network to carry the traffic opaquely across the lower-layer network resources. A transport network may be constructed from equipment utilizing any of a number of different transport technologies such as the optical transport infrastructure (Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH) and Optical Transport Network (OTN)) or packet transport as epitomized by the MPLS Transport Profile (MPLS-TP).

All transport networks have high benchmarks for reliability and operational simplicity. This suggests a common, technology-independent management/control paradigm that can be extended to represent and configure specific technology attributes.

This document describes the high-level requirements facing transport networks in order to provide open interfaces for resource programmability and control/management automation. Furthermore, gap analysis against existing models are also provided so that it can used as the guidance to separate efforts/drafts proposing new models or augmentation models based on existing models.

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 4, 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. Scope .......................................................... 4
3. High-level Modeling Requirements .............................. 5
   3.1. Generic Requirements ..................................... 5
   3.2. Transport Network and TE Topology Requirements ........... 6
      3.2.1. Topological Link Requirements ....................... 6
      3.2.2. Topology Node Requirements ........................... 6
      3.2.3. Termination Point Requirements ....................... 6
   3.3. Transport Service Requirements ............................ 7
   3.4. Tunnel/LSP Requirements ................................... 7
4. Scenarios ....................................................... 7
   4.1. Single-domain Scenario ................................... 7
   4.2. Multi-domain Scenario .................................... 9
   4.3. Multi-layer Scenario ..................................... 11
   4.4. Function Summary and Related YANG Models ................ 11
5. Function Gap Analysis on YANG Model Level .................... 12
   5.1. Topology Related Functions ............................... 12
      5.1.1. Obtaining Access Point Info ........................... 13
      5.1.2. Obtaining Topology ................................... 13
      5.1.3. Virtual Network Operations ........................... 13
   5.2. Tunnel Operations .......................................... 14
   5.3. Service Requests ........................................... 14
6. IANA Considerations ............................................ 14
7. Security Considerations ....................................... 14
1. Introduction

A transport network is a server-layer network designed to provide connectivity services, or more advanced services like Virtual Private Networks (VPN) for a client-layer network to carry the client traffic opaquely across the server-layer network resources. It acts as a pipe provider for upper-layer networks, such as IP network and mobile networks.

Transport networks, such as Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH), Optical Transport Network (OTN), Wavelength Division Multiplexing (WDM), and flexi-grid networks, are often built using equipments from a single vendor and are managed using proprietary interfaces to dedicated Element Management Systems (EMS) / Network Management Systems (NMS). All transport networks have high benchmarks for reliability and operational simplicity. This suggests a common, technology-independent management/control paradigm that is extended to represent and configure specific technology attributes.

Network providers need a common way to manage multi-vendor and multi-domain transport networks (where each domain is an island of equipments from a single supplier) and this requirement has been further stressed by the expansion in network size. At the same time, applications such as data center interconnection require larger and more dynamic connectivities. Therefore, transport networks face new challenges going beyond automatic provisioning of tunnel setup enabled by GMPLS (Generalized Multi-Protocol Label Switching) protocols to achieve automatic service provisioning, as well as address opportunities enabled by partitioning the transport network through the process of resource slicing. With a reduction in operational expenditure (OPEX) and capital expenditure (CAPEX) as the usual objectives, a common interface to transport network controllers are considered by network providers as a way to meet the requirements. The concept of Software Defined Networking (SDN) leverages these ideas.

The YANG language [RFC6020] is currently the data modeling language of choice within the IETF and has been adopted by a number of industry-wide open management and control initiatives. YANG may be
used to model both configuration and operational states; it is vendor-neutral and supports extensible APIs for control and management of elements.

This document first specifies the scope and provides high-level requirements for transport network open interface modelling. Furthermore, detailed gap analysis of the typical scenarios with the existing model are provided. Thus, this document can used as a reference of existing models, and provides information of the missing ones which suggest further work.

2. Scope

For this draft, we use the domain controller as the reference point, with South Bound Interface (SBI) to the transport devices and North Bound Interface (NBI) to the orchestrator.

Transport networks have been evolving and deploying for decades, making them very heterogeneous. New and legacy transport devices support many protocols such as Path Computation Element Protocol (PCEP), TL1, SNMP, CLI, XML, NETCONF, Openflow etc. Domain controllers interfacing with transport devices need to support these protocols on its SBI, making the southbound fragmented. Domain controllers abstract the fragmented southbound view for its northbound clients by normalizing the NBI across various technologies, protocols, and vendors. The focus of this document is not to go into various southbound protocols to interface with the transport devices. Instead, this document focuses on the models that can be used by the domain controller and the orchestrator for various use cases identified in later sections of this document.

There is an ongoing unofficial weekly meeting among a group of individuals (see the github files [Transport-modeling-github] for more meeting minutes and materials produced), focusing on the efforts of analyzing the IETF models against a list of well known use cases to identify gaps. This document captures this efforts and summarized the main work and key findings of this group work.

YANG models are currently developed not only in IETF, but also in other Standard Development Organizations (SDO) such as ONF and MEF, which can be used on the interfaces of a domain controller and an orchestrator. Each domain controller and orchestrator can use models developed by different SDOs. Therefore it is important to ensure that deployment use cases and related functionalities are supported by all models to allow a seamless translation/mediation between systems using different models.
If the Abstraction and Control of Traffic-Engineered Networks (ACTN) defined in [I-D.ceccarelli-teas-actn-framework] is used as a reference architecture, then the focus is equivalent to MPI (MDSC-PNC Interface) and CMI (CNC-MDSC Interface). More details about the relationship of the type of models and the type of ACTN interfaces can be found in [I-D.zhang-teas-actn-yang].

3. High-level Modeling Requirements

This section covers various high-level modeling requirements for transport networks.

3.1. Generic Requirements

The following are generic requirements for transport models:

- **User Intent**: Transport models should maintain separation between high level user intent and the operational state of the network. For example, maintaining separation between user service request, including all constraints, and the actual service and connection state in the network.

- **State Management**: Network and service objects should support the following states: administrative state, operational state, and lifecycle state. Administrative state and operational states are well understood. Lifecycle state is defined in the ONF and it is used to track the planned deployment allocation of the related entity in the model as well as withdrawal of resources. Here the lifecycle state includes planned state, potential state, installed state, and pending removal state.

- **Identifiers**: Network and service objects should support the following identifier:
  - **ID**: A unique entity ID provided by the controller. The identifier SHOULD be chosen such that the same entity in a real network topology will always be identified through the same ID, even if the model is instantiated in separate datastores. Controller may choose to capture semantics in the identifier, for example to indicate the type of entity and/or the type of the parent identity.
  - **Name**: A unique name provided by the client for the entity. The name can be modified, if required, by the client.
  - **User Labels**: A list of freeform strings that can be used as alias for the entity by the client. Multiple user labels are...
permitted for the entity, and client can edit these user labels. User labels do not need to be unique.

3.2. Transport Network and TE Topology Requirements

3.2.1. Topological Link Requirements

The model should support the following Topological Links:

- Physical Links
- Abstract Links [RFC7926]
- Compound Link which are internally aggregated lower level links
- Access Links which connect the router port to the client port of the transport system
- Transitional Links which provide adaptation capability between layers within a network element

The Link should support various link related attributes like cost, latency, capacity, risk characteristics (including shared risk). The model should provide clear association between Link and its topology (including virtual topology), nodes and termination points.

The model should provide association between the link and any underlay circuit / service supporting the Link.

3.2.2. Topology Node Requirements

The model should support the following Topology Node:

- Physical Node
- Abstract Node
- Chassis / Forwarding Domain

[Editors’ note: more details will be added later, which can be found in [Transport-requirements-github].]

3.2.3. Termination Point Requirements

[Editors’ note: this will be added later, which can be found in [Transport-requirements-github].]
3.3. Transport Service Requirements

[Editors’ note: this will be added later, which can be found in [Transport-requirements-github].]

3.4. Tunnel/LSP Requirements

[Editors’ note: this will be added later which can be found in [Transport-requirements-github].]

4. Scenarios

There are several scenarios (a.k.a., use cases) where an open interface via domain controller to access server-layer (transport) network resources would be useful. Three scenarios are provided and can be used for model instantiation exercise to identify missing pieces of existing models. Note the models provided in this draft is for explanation purpose, the group effort actually uses slight different network examples for gap analysis exercise (see [Transport-usecases-github] for more details).

4.1. Single-domain Scenario

The first scenario is depicted as below (Figure 1):
Figure 1: Scenario 1: Data centers interconnected via a transport network and the controller architecture

(a) Data Centers interconnected via a transport network

(b) The controller architecture for data center interconnection
For the data center operator, as a client of the transport network, assuming the objective is to trigger the transport network to provide connectivity on demand, the following capabilities, at a minimum, would be required on the common interface between the two controllers illustrated in Figure 1:

- The ability to obtain information about a set of access points of the transport network, including information such as access point identifiers, capabilities, etc.; for instance, transport-network-side end point identifiers related to the access link between DC1 and Transport NE A.

- The capability to send a request for a service using the aforementioned access point information, as well as the ability to retrieve a list of service requests and their status. In this request, it should at least be possible to include source node, destination node, and requested bandwidth to request the transport network to set up tunnels/paths so as to provide the requested connectivity for the service request.

- Note that in this case, the acquisition of the topology, be it physical or logical, of the transport network is not a compulsory requirement, but it may indeed be able to give data center providers more control over the transport resource usage. Furthermore, the client controller can impose a virtual network of its own choice by requesting a slice of network resource with its choice of network parameters (such as network topology type, bandwidth etc.).

4.2. Multi-domain Scenario

The second scenario, more complicated than the first, is depicted as below (Figure 2). In this example, we focus on the management and control via common interfaces for multi-domain networks with homogeneous technologies (such as OTN), but it can be extended further to multi-domain networks with heterogeneous technologies with higher complexity.
Figure 2: Scenario 2: Multi-domain network control and management

For the second scenario, the orchestrator controls and manages three distinct network domains, each controlled/managed by their domain controller. This scenario is of interest not only to transport-only
networks, but also to heterogenous network orchestration such as coordinating the transport, the radio (5G) and packet core domains. But to keep the functions explanation later accurate, only transport-only multi-domain networks are considered.

In order to orchestrate across domains/layers, besides the capabilities mentioned for the first scenario, the orchestrator needs its interface between domain controllers to be equipped with the following additional functions:

- Access to the topologies reported by each domain controller, including cross-domain links for the purpose of planning and requesting the paths of end-to-end tunnels. Depending on the abstraction level of the reported topology, the orchestrator has different control granularities.
- Alternatively, the capability for the orchestrator to request "path computation" to a domain controller in order to create an end-to-end tunnel stitched together by different connection contribution obtained by consulting to each domain controller.
- The ability to set up, delete and modify tunnels, be it within one domain or across multiple domains. Furthermore, it should have the ability to view the tunnels created within each domain as well as those that cross domains as reported by each domain controller.

4.3. Multi-layer Scenario

In the first use case, if there are multiple technologies involved, then it can be considered as a multi-layer case. [Editors’ note: more details to be added later, some can be found in [Transport-usecases-github].]

4.4. Function Summary and Related YANG Models

For the common interface of a transport controller towards a northbound client, six main functions are derived from the scenarios explained in the last section. They are summarized in the table below and we also match these functions with YANG models that are being developed in existing drafts.
<table>
<thead>
<tr>
<th>Functions</th>
<th>Description</th>
<th>Related Existing YANG Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obtaining Access</td>
<td>Getting the necessary access points info</td>
<td>ietf-network.yang</td>
</tr>
<tr>
<td>Point Info</td>
<td></td>
<td>ietf-network-topology.yang</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ietf-te-topology.yang</td>
</tr>
<tr>
<td>Obtaining Topology</td>
<td>Getting the topology info</td>
<td>ietf-te-topology.yang</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ietf-otn-topology.yang</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ietf-wson-topology.yang</td>
</tr>
<tr>
<td>Tunnel Operations</td>
<td>Tunnel Setup, Deletion Modification and Info</td>
<td>ietf-te.yang</td>
</tr>
<tr>
<td></td>
<td>Retrieval</td>
<td>ietf-otn-tunnel.yang</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Request</td>
<td>Requesting connectivity service and retrieval</td>
<td>ietf-transport-service.yang</td>
</tr>
<tr>
<td></td>
<td>the list of service</td>
<td>ietf-actn-vn.yang</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path Comp.</td>
<td>Path Computation pre service provisioning</td>
<td>ietf-te.yang</td>
</tr>
<tr>
<td>Virtual Network</td>
<td>Requesting a virtual network and related control</td>
<td>ietf-te-topology.yang</td>
</tr>
<tr>
<td>Operations</td>
<td>operations, (e.g., update, deletion)</td>
<td>ietf-actn-vn.yang</td>
</tr>
</tbody>
</table>

Analysis and descriptions of whether and how these functions are supported by the YANG models are provided in more detail in Section 5.

5. Function Gap Analysis on YANG Model Level

5.1. Topology Related Functions

As shown in the previous section, the functions of obtaining access point information, obtaining topology, and imposing virtual network operations can take advantages of the same set of topology YANG models. These functions are briefly explained further in the following sub-sections.
5.1.1. Obtaining Access Point Info

For cases such as scenario 1, a client may have no interest in directly controlling network resources, but might want an automated common control interface for initiating service requests. In this case, a transport domain controller may provide the access point information. This information can then be used in service request sent over the common interface.

The TE Topology YANG model provided in [TE-topo] [I-D.ietf-teas-yang-te-topo] can be used to provide a list of links. If the remote node and termination point information is unknown, it is omitted from the reported information. If the client-side node and termination point information is obtained via configuration or a distributed discovery mechanism, then it can also be added into the reported information. Technology-specific details might also be needed to further express the constraints/attributes associated with the access points. Note that all of this information is usually read only.

5.1.2. Obtaining Topology

Refer to [I-D.ietf-teas-yang-te-topo] for explanations and examples on how to obtain the topology. For technology specific topology information, other models such as those provided in [WDM-Topo] [I-D.ietf-ccamp-wson-yang] and [ODU-Topo] [I-D.zhang-ccamp-l1-topo-yang] may be used.

There are two ways provided in [I-D.ietf-teas-yang-te-topo] in terms of how to present a multi-layer topology, discussions have been carried out among the unofficial group in terms of how the transitional link approach can work and the discussion material will be available soon in the github [Transport-modeling-github].

5.1.3. Virtual Network Operations

There are two ways to request the creation of a virtual network. One is to define the topology explicitly using the model provided in the topology YANG drafts listed in previous section. The other way is to provide an estimated traffic information (a traffic matrix) and ask for a domain controller of the provider network to provide a virtual network that can fulfill the demand. This second approach is supported by the YANG model in [I-D.lee-teas-actn-vn-yang].
5.2. Tunnel Operations

The current [TE-Tunnel] [I-D.ietf-teas-yang-te] provides a technology agnostic Traffic-Engineered (TE) to manage and configure tunnel. The model included in that draft is currently being developed to make it generic for both controller and device usage. In the latest version, it already provides such a generic TE tunnel model that can cater to the base requirements for tunnel operations but it may need to be augmented to support controller-specific operations.

Furthermore, technology-specific augmentations of the base generic TE tunnel models are needed. For example, for Optical Channel (OCh) (note: ITU is updating this term as OTSi.) tunnels in WDM networks, information such as the lambda resource usage is needed. Similarly, for ODU tunnels, information such as ODU-specific client signal, tributary slot information etc. is needed.

For path computation, [I-D.busibel-teas-yang-path-computation] presents now only use cases but YANG model work is also under consideration to provide stateless path computation RPC. There is currently ongoing discussions on how to provide such a function using the TE tunnel model defined in [I-D.ietf-teas-yang-te] as a base.

5.3. Service Requests

Service model is an important type of models, such as the one provided in [I-D.zhang-teas-transport-service-model], that enables automated operations between a client controller or an orchestrator and a domain controller. This transport connectivity service model is different from the model of a tunnel since the transport connectivity service model are enforced over the client-server interfaces, and it hides unnecessary provider network details from a client.

6. IANA Considerations

This document requests no IANA actions.

7. Security Considerations

Clearly modifying server-layer resources will have a significant impact on network infrastructure. More specifically they will provide the services and applications running across client-layers, which the server-layer is supporting. Therefore, security must be an important consideration when implementing the architecture, models and protocol mechanisms discussed in this document.
Communicating service and network information (including access point identifiers, capabilities, topologies, etc.) across external interfaces represents a security risk. Thus, mechanisms to encrypt or preserve the domain topology confidentiality should be used.

A key consideration are the external protocols (those shown as entering or leaving the orchestrator and controllers shown in Figure 2 (Scenario 2: Multi-domain network control and management)) which must be appropriately secured. This security should include authentication and authorization to control access to different functions that the orchestrator may perform to modify or create state in the server-layer, and the establishment and management of the orchestrator to controller relationship.

The orchestrator will contain significant data about the network domains, the services carried by each domain, and customer type information. Therefore, access to information held in the orchestrator must be secured. Since such access will be largely through external mechanisms, it may be pertinent to apply policy-based controls to restrict access and functions.

8. Manageability Considerations

The core objectives of this document are to assist in the deployment and operation of transport services across server-layer network infrastructure. The model-driven management/control principles, which are vendor-neutral and supported by extensible APIs, should be utilized.

The open models described in this document are based on YANG [RFC6020] and the RESTCONF [RESTCONF] messaging protocol, a REST-like protocol running over HTTP for accessing data defined in YANG, may also be used.

9. Acknowledgements

We would like to thank Young Lee, Igor Bryskin and Aihua Guo for their comments and discussions.

10. Contributing Authors

The following people all contributed to this document and are listed below:

Ruiquan Jing
China Telecom
Email: jingrq@ctbri.com.cn
11. References

11.1. Normative References

[I-D.ietf-netconf-restconf]


11.2. Informative References

[I-D.busibel-teas-yang-path-computation]

[I-D.ceccarelli-teas-actn-framework]
[I-D.ietf-ccamp-wson-yang]
Lee, Y., Dhody, D., Zhang, X., Guo, A., Lopezalvarez, V.,
King, D., and B. Yoon, "A Yang Data Model for WSON Optical
Networks", draft-ietf-ccamp-wson-yang-01 (work in
progress), April 2016.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Chen,
X., Jones, R., and B. Wen, "A YANG Data Model for Traffic
Engineering Tunnels and Interfaces", draft-ietf-teas-yang-
te-02 (work in progress), October 2015.

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and
O. Dios, "YANG Data Model for TE Topologies", draft-ietf-
teas-yang-te-topo-04 (work in progress), March 2016.

[I-D.lee-teas-actn-vn-yang]
Lee, Y., Ceccarelli, D., Miyasaka, T., Park, P., and B.
Yoon, "A Yang Data Model for ACTN VN Operation", draft-
lee-teas-actn-vn-yang-01 (work in progress), July 2016.

[I-D.zhang-ccamp-l1-topo-yang]
Zhang, X., Rao, B., and X. Liu, "A YANG Data Model for
Layer 1 Network Topology", draft-zhang-ccamp-l1-topo-
yang-01 (work in progress), December 2015.

[I-D.zhang-teas-actn-yang]
Lee, Y., Zhang, X., Yoon, B., and O. Dios, "Applicability
of YANG models for Abstraction and Control of Traffic
Engineered Networks", draft-zhang-teas-actn-yang-01 (work
in progress), October 2016.

[I-D.zhang-teas-transport-service-model]
Zhang, X. and J. Ryoo, "A Service YANG Model for
Connection-oriented Transport Networks", draft-zhang-teas-
transport-service-model-00 (work in progress), July 2016.

[RFC7926]  Farrel, A., Ed., Drake, J., Bitar, N., Swallow, G.,
Ceccarelli, D., and X. Zhang, "Problem Statement and
Architecture for Information Exchange between
Interconnected Traffic-Engineered Networks", BCP 206,
RFC 7926, DOI 10.17487/RFC7926, July 2016,

[Transport-modeling-github]
"https://github.com/TransportModels/IETF-Transport-
[Transport-requirements-github]

[Transport-usecases-github]

Authors’ Addresses

Xian Zhang (editor)
Huawei Technologies
F3-5-B R&D Center, Huawei Industrial Base, Bantian, Longgang District
Shenzhen, Guangdong  518129
P.R.China

Email: zhang.xian@huawei.com

Anurag Sharma (editor)
Infinera
US

Email: ansharma@infinera.com

Sergio Belotti
Nokia
Italy

Email: sergio.belotti@nokia.com

Tara Cummings
Ericsson

Email: tara.cummings@ericsson.com