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 September 27, 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.
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

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. Requirements Language ................................. 3
3. Terminology and Notation ............................... 3
   3.1. Tree Diagrams .................................. 4
   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. Acknowledgements .................................... 102
14. References .......................................... 103
   14.1. Normative References .......................... 103
   14.2. Informative References ....................... 106
Appendix A. Example ...................................... 108
Appendix B. Contributor Addresses ....................... 111
Authors’ Addresses ...................................... 112
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).
- Error message (PCErr).
- Request Parameters object (RP).
- 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>tls-server</td>
<td>ietf-tls-server</td>
<td>[I-D.ietf-netconf-tls-client-server]</td>
</tr>
<tr>
<td>tls-client</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 RSVP for LSP Tunnels</td>
<td>[RFC3209]</td>
</tr>
<tr>
<td>OSPF Protocol Extensions for Path Computation</td>
<td>[RFC5088]</td>
</tr>
<tr>
<td>IS-IS Protocol Extensions for Path Computation</td>
<td>[RFC5089]</td>
</tr>
<tr>
<td>Path Computation Element (PCE) Discovery</td>
<td>[RFC5440]</td>
</tr>
<tr>
<td>Path Computation Element (PCE) Communication</td>
<td></td>
</tr>
<tr>
<td>Preserving Topology</td>
<td>[RFC5520]</td>
</tr>
</tbody>
</table>
Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism
Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)

[RFC5541] [RFC5557] [RFC6991] [RFC8177] [RFC8231] [RFC8232] [RFC8253] [RFC8281] [RFC8306] [RFC8341] [RFC8408] [RFC8408]
### 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
      +--rw `speaker-entity-id?` string
      |     |   {stateful-sync-opt}? (stateful-sync-opt)?
      +--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]
      |   |   ...
      |   +--ro `path-keys {path-key}?`
      |   |   +--ro `path-keys*` [path-key]
      |   |   |
      |   +--rw `peers`
      |   +--rw `peer*` [addr]
      |
      +--ro `sessions`
rpocs:
  +---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? 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]
      +--ro plsp-id -> /pcep/entity/lsp-db/lsp/plsp-id
      +--ro pcc-id -> /pcep/entity/lsp-db/lsp/pcc-id
    +--ro lsp* [plsp-id pcc-id]
      +--ro plsp-id uint32
      +--ro pcc-id inet:ip-address
    +--ro lsp-ref
      +--ro source?
      |   -> /te:te/lsps-state/lsp/source
      +--ro destination?
      |   -> /te:te/lsps-state/lsp/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
Internet-Draft                  PCE-YANG                      March 2019

|    +--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
|        -> /pcep/entity/lsp-db/association-list/type
|    +--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)
|                +--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
|                       ...

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 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}?
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 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
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}?
++-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-mags? 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* uint64 {sync-opt}?
    [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]
      ++-ro plsp-id -> /pcep/entity/lsp-db/lsp/plsp-id
      ++-ro pcc-id -> /pcep/entity/lsp-db/lsp/pcc-id
    ++-ro lsp* [plsp-id pcc-id]
      ++-ro plsp-id uint32
      ++-ro pcc-id inet:ip-address
      ++-ro lsp-ref
        ++-ro source?
|   | -> /te:te/lsps-state/lsp/source
|   | +--ro destination?
|   | | -> /te:te/lsps-state/lsp/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
|   | | -> /pcep/entity/lsp-db/association-list/type
|   | +--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)
+--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)
          +--rw hex-key-string?
            key-chain: hex-key-string?
            +--rw hexadecimal-string? yang:hex-string
    +--:(auth-tls) {tls}?
      +--rw (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
    +--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 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
      |   +++--ro peer-addr?    -> /pcep/entity/peers/peer/addr
      |   +++--ro session-initiator?
      |     |   -> /pcep/entity/peers/peer/sessions/session/initiator
      |   +++--ro overloaded?   boolean
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-corrrupt-rcvd?  yang:counter32
      ++--ro num-req-sent?       yang:counter32
      ++--ro num-req-sent-pend-rep? yang:counter32
      ++--ro num-req-sent-ero-rcvd? yang:counter32
      ++--ro num-req-sent-nopath-rcvd? yang:counter32
      ++--ro num-req-sent-cancel-rcvd? yang:counter32
```
+--ro num-req-sent-error-rcvd?    yang:counter32
+--ro num-req-sent-timeout?       yang:counter32
+--ro num-req-sent-cancel-sent?   yang:counter32
+--ro num-req-rcvd?               yang:counter32
+--ro num-req-rcvd-pend-rep?      yang:counter32
+--ro num-req-rcvd-ero-sent?      yang:counter32
+--ro num-req-rcvd-nopath-sent?   yang:counter32
+--ro num-req-rcvd-cancel-sent?   yang:counter32
+--ro num-req-rcvd-error-sent?    yang:counter32
+--ro num-req-rcvd-cancel-rcvd?   yang:counter32
+--ro num-rep-rcvd-unknown?       yang:counter32
+--ro num-req-rcvd-unknown?       yang:counter32
+--ro svec {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
augment /pcep:pcep/pcep:entity/pcep:peers/pcep:peer/pcep:sessions 
/pcep:session: 
+--ro pcep-stats 
  +--ro avg-rsp-time?    uint32
  +--ro lwm-rsp-time?    uint32
+--ro hwm-rsp-time?             uint32
+--ro num-pcreq-sent?           yang:counter32
+--ro num-pcreq-rcvd?           yang:counter32
+--ro num-pcrep-sent?           yang:counter32
+--ro num-pcrep-rcvd?           yang:counter32
+--ro num-pcerr-sent?           yang:counter32
+--ro num-pcerr-rcvd?           yang:counter32
+--ro num-pcntf-sent?           yang:counter32
+--ro num-pcntf-rcvd?           yang:counter32
+--ro num-keepalive-sent?       yang:counter32
+--ro num-keepalive-rcvd?       yang:counter32
+--ro num-unknown-rcvd?         yang:counter32
+--ro num-corrupt-rcvd?         yang:counter32
+--ro num-req-sent?             yang:counter32
+--ro num-req-sent-pend-rep?    yang:counter32
+--ro num-req-sent-ero-rcvd?    yang:counter32
+--ro num-req-sent-nopath-rcvd? yang:counter32
+--ro num-req-sent-cancel-rcvd? yang:counter32
+--ro num-req-sent-error-rcvd?  yang:counter32
+--ro num-req-sent-timeout?     yang:counter32
+--ro num-req-sent-cancel-sent? yang:counter32
+--ro num-req-rcvd?             yang:counter32
+--ro num-req-rcvd-pend-rep?    yang:counter32
+--ro num-req-rcvd-ero-sent?    yang:counter32
+--ro num-req-rcvd-nopath-sent? yang:counter32
+--ro num-req-rcvd-cancel-sent? yang:counter32
+--ro num-req-rcvd-error-sent?  yang:counter32
+--ro num-req-rcvd-cancel-rcvd? yang:counter32
+--ro num-rep-rcvd-unknown?     yang:counter32
+--ro num-req-rcvd-unknown?     yang:counter32
  | +--ro 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-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
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/peer)
- Path-Key
- OF
- GCO
- P2MP
- GMPLS
- Inter-Layer
- Stateful PCE
- Segement Routing
- 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@2019-03-26.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 netconf-acm;
        reference
            "RFC 8341: Network Configuration Protocol (NETCONF) Access Control Model";
    }
    import ietf-tls-server {
        prefix tls-server;
    }

reference
"[I-D.ietf-netconf-tls-client-server]: YANG Groupings for TLS
Clients and TLS Servers";
}
import ietf-tls-client {
  prefix tls-client;
  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
(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 2019-03-26 {
  description
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
        "The operational status of the LSP";
    reference
}
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
 */

class feature svec {
  description
    "Support synchronized path computation.";
  reference
    "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)";
}

class feature gmpls {
  description
    "Support GMPLS.";
}

"RFC 8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE";
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

//info

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 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 {
        bit gmpls {
          if-feature "gmpls";
          description
            "Path computation with GMPLS link constraints.";
        }
        bit bi-dir {
          description
            "Bidirectional path computation.";
        }
      }
    }
  }
}
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 capabilites"
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-routinig]: 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 OF";
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.";
        }
    }
}

reference
"RFC 5541: Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCE)";
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 PECP 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
do 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";
}
carrier of-list {

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";
    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";
    }
}
list lsp {
    key "plsp-id pcc-id";
    description
        "List of all LSPs in LSP-DB";
    leaf plsp-id {
        type uint32 {
            range "1..1048575";
        }
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 "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 "Tunnel identifier used in the SESSION that remains constant over the life of the tunnel.";
  }
}
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
    "RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels";
}
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
    "RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels";
}
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 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";
    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";
    }
    description
    "Reference to PCC peer address which
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 {
      netconf-acm:default-deny-all;
      type string;
      description
      "Key string in ASCII format.";
    }
  }
  case hexadecimal {
    if-feature "key-chain:hex-key-string";
    leaf hexadecimal-string {
      netconf-acm: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 tls-server:tls-server-grouping {
            description
            "Server TLS information.";
          }
          description
          "TLS related information";
        }
      }
    }
  }
}
case client {
    container tls-client {
        uses tls-client: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;
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
  "RFC 5440: Path Computation Element (PCE)
  Communication Protocol (PCE)";
}
leaf remote-id {
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 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-dead-timer {
  type uint8;
  units "seconds";
  description "The peer’s dead-timer interval for this PCEP 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)";
}

error-message "Invalid Peer keepalive timer";
]
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 (PCE)"
}
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."

reference
"RFC 5440: Path Computation Element (PCE) Communication Protocol (PCE)"

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 '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 5440: Path Computation Element (PCE) Communication Protocol (PCE)"
"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
//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 (PCEP)"
};

//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
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 resynchronization 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
    synchronzation is re-triggered for all LSPs from
    the PCC. The rpc on the PCC will be ignored.";
}
//input
}
//rpc

<CODE ENDS>

10.2. ietf-pcep-stats module

<CODE BEGINS> file "ietf-pcep-stats@2019-03-26.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";
    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 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 (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 2019-03-26 {
  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 average response time. If an average response time has not been calculated 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
        "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’" {
  description
  "Valid for PCEP Peer as PCC";
}
type yang:counter32;
description
  "The number of PCRep messages sent.";
}
leaf num-pcrep-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 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 {
  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";
}
type yang:counter32;
description
"The number of requests that have been sent to a peer and have been abandoned because the peer has taken too long to respond to them.";

leaf num-req-sent-cancel-sent {
  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 explicitly cancelled by the local PCEP entity sending a PCNtf.";

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";
}
+ "../..//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 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-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";
    }
}

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";
  }
  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.";
}
container initiation {
  if-feature "pcep:pce-initiated"
  description
  "PCE-Initiated related statistics";
leaf num-pcinitiate-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 PCInitiate messages sent.";
}

leaf num-pcinitiate-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 PCInitiate messages received.";
}

leaf num-initiate-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 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 {
  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 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 {
  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.";
}

/*stateful

container 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 attempts to expand a duplicated path-key.";
  }
}*/
"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
/*
 * 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.

```yml
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.";
}
```

Dhody, et al. Expires September 27, 2019 [Page 100]
11. Security Considerations

The YANG module 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 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 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:

/pcep/lsp-db - All the LSPs in the network. Unauthorized access to this could provide the all path and network usage information.
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 this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

- **trigger-resync** - trigger resynchronization with the PCE. Unauthorized access to this could force a PCEP session into continuous state synchronization.

### 12. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

**URI:** urn:ietf:params:xml:ns:yang:ietf-pcep

**Registrant Contact:** The PCE WG of the IETF.

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

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

- **Name:** ietf-pcep
- **Namespace:** urn:ietf:params:xml:ns:yang:ietf-pcep
- **Prefix:** pcep
- **Reference:** [This.I-D]

- **Name:** ietf-pcep-stats
- **Namespace:** urn:ietf:params:xml:ns:yang:ietf-pcep-stats
- **Prefix:** pcep-stats
- **Reference:** [This.I-D]

### 13. 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.
14. References

14.1. Normative References


14.2. Informative References


Appendix A. Example

The example below provide an overview of PCEP peer session informations and LSP-DB in the Yang Module.

+-------+                  +-------+
|       |                  |       |
| PCC1  |<---------------->|       |
+-------+                  |       |
IP:192.0.2.1                |       |
|  PCE  |                  |
|       |                  |       |
+-------+                  |       |
IP:192.0.2.2                |       |
|       |                  |
| PCC4  |<---------------->|       |
+-------+                  |       |
IP:2001:DB8::4              |       |

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,
        }
    ]
]
"pcc-id": "192.0.2.1",
"lsp-ref": {
  "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",
  "lsp-ref": {
    "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,
      }\n    }
  }
]
{
    "peer": {
        "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  
Jabil  
8281 Greensboro Drive, Suite 200  
McLean VA 22102  
USA  
EMail: Xufeng_Liu@jabil.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: udayasreereddy@gmail.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