

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

D. Dhody, Ed.  
Huawei Technologies  
J. Hardwick  
Metaswitch  
V. Beeram  
Juniper Networks  
J. Tantsura  
October 27, 2016

**A YANG Data Model for Path Computation Element Communications Protocol  
(PCEP)  
draft-ietf-pce-pcep-yang-01**

**Abstract**

This document defines a YANG data model for the management of Path Computation Element communications Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs. The data model includes configuration data and state data (status information and counters for the collection of statistics).

**Status of This Memo**

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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

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

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

**Copyright Notice**

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

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Requirements Language . . . . .	<a href="#">3</a>
<a href="#">3.</a>	Terminology and Notation . . . . .	<a href="#">3</a>
<a href="#">3.1.</a>	Tree Diagrams . . . . .	<a href="#">4</a>
<a href="#">3.2.</a>	Prefixes in Data Node Names . . . . .	<a href="#">5</a>
<a href="#">4.</a>	Objectives . . . . .	<a href="#">5</a>
<a href="#">5.</a>	The Design of PCEP Data Model . . . . .	<a href="#">6</a>
<a href="#">5.1.</a>	The Entity . . . . .	<a href="#">19</a>
<a href="#">5.2.</a>	The Peer Lists . . . . .	<a href="#">19</a>
<a href="#">5.3.</a>	The Session Lists . . . . .	<a href="#">20</a>
<a href="#">5.4.</a>	Notifications . . . . .	<a href="#">20</a>
<a href="#">6.</a>	Advanced PCE Features . . . . .	<a href="#">20</a>
<a href="#">6.1.</a>	Stateful PCE's LSP-DB . . . . .	<a href="#">21</a>
<a href="#">7.</a>	Open Issues and Next Step . . . . .	<a href="#">21</a>
<a href="#">7.1.</a>	The PCE-Initiated LSP . . . . .	<a href="#">21</a>
<a href="#">7.2.</a>	PCEP over TLS (PCEPS) . . . . .	<a href="#">21</a>
<a href="#">8.</a>	PCEP YANG Module . . . . .	<a href="#">22</a>
<a href="#">9.</a>	Security Considerations . . . . .	<a href="#">94</a>
<a href="#">10.</a>	Manageability Considerations . . . . .	<a href="#">94</a>
<a href="#">10.1.</a>	Control of Function and Policy . . . . .	<a href="#">94</a>
<a href="#">10.2.</a>	Information and Data Models . . . . .	<a href="#">94</a>
<a href="#">10.3.</a>	Liveness Detection and Monitoring . . . . .	<a href="#">94</a>
<a href="#">10.4.</a>	Verify Correct Operations . . . . .	<a href="#">94</a>
<a href="#">10.5.</a>	Requirements On Other Protocols . . . . .	<a href="#">94</a>
<a href="#">10.6.</a>	Impact On Network Operations . . . . .	<a href="#">94</a>
<a href="#">11.</a>	IANA Considerations . . . . .	<a href="#">94</a>
<a href="#">12.</a>	Acknowledgements . . . . .	<a href="#">95</a>
<a href="#">13.</a>	References . . . . .	<a href="#">95</a>
<a href="#">13.1.</a>	Normative References . . . . .	<a href="#">95</a>
<a href="#">13.2.</a>	Informative References . . . . .	<a href="#">96</a>
<a href="#">Appendix A.</a>	Contributor Addresses . . . . .	<a href="#">98</a>
	Authors' Addresses . . . . .	<a href="#">99</a>

## [1.](#) Introduction

The Path Computation Element (PCE) defined in [[RFC4655](#)] is an entity that is capable of computing a network path or route based on a network graph, and applying computational constraints. A Path



Computation Client (PCC) may make requests to a PCE for paths to be computed.

PCEP is the communication protocol between a PCC and PCE and is defined in [RFC5440]. PCEP interactions include path computation requests and path computation replies as well as notifications of specific states related to the use of a PCE in the context of Multiprotocol Label Switching (MPLS) and Generalized MPLS (GMPLS) Traffic Engineering (TE). [I-D.ietf-pce-stateful-pce] specifies extensions to PCEP to enable stateful control of MPLS TE LSPs.

This document defines a YANG [RFC6020] data model for the management of PCEP speakers. It is important to establish a common data model for how PCEP speakers are identified, configured, and monitored. The data model includes configuration data and state data (status information and counters for the collection of statistics).

This document contains a specification of the PCEP YANG module, "ietf-pcep" which provides the PCEP [RFC5440] data model.

## **2. Requirements Language**

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

## **3. Terminology and Notation**

This document uses the terminology defined in [RFC4655] and [RFC5440]. In particular, it uses the following acronyms.

- o Path Computation Request message (PCReq).
- o Path Computation Reply message (PCRep).
- o Notification message (PCNtf).
- o Error message (PCErr).
- o Request Parameters object (RP).
- o Synchronization Vector object (SVEC).
- o Explicit Route object (ERO).

This document also uses the following terms defined in [RFC7420]:

- o PCEP entity: a local PCEP speaker.



- o PCEP peer: to refer to a remote PCEP speaker.
- o PCEP speaker: where it is not necessary to distinguish between local and remote.

Further, this document also uses the following terms defined in [\[I-D.ietf-pce-stateful-pce\]](#) :

- o Stateful PCE, Passive Stateful PCE, Active Stateful PCE
- o Delegation, Revocation, Redelegation
- o LSP State Report, Path Computation Report message (PCRpt).
- o LSP State Update, Path Computation Update message (PCUpd).

[\[I-D.ietf-pce-pce-initiated-lsp\]](#) :

- o PCE-initiated LSP, Path Computation LSP Initiate Message (PCInitiate).

[\[I-D.ietf-pce-lsp-setup-type\]](#) :

- o Path Setup Type (PST).

[\[I-D.ietf-pce-segment-routing\]](#) :

- o Segment Routing (SR).

### **[3.1.](#) Tree Diagrams**

A graphical representation of the complete data tree is presented in [Section 5](#). The meaning of the symbols in these diagrams is as follows and as per [\[I-D.ietf-netmod-rfc6087bis\]](#):

- o Brackets "[" and "]" enclose list keys.
- o Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- o Symbols after data node names: "?" means an optional node, "!" means a presence container, and "\*" denotes a list and leaf-list.
- o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- o Ellipsis ("...") stands for contents of subtrees that are not shown.



### 3.2. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
te	ietf-te	[I-D.ietf-teas-yang-te]
te-types	ietf-te-types	[I-D.ietf-teas-yang-te]
key-chain	ietf-key-chain	[I-D.ietf-rtgwg-yang-key-chain]

Table 1: Prefixes and corresponding YANG modules

## 4. Objectives

This section describes some of the design objectives for the model:

- o 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.
- o The data model should be suitable for new implementations to use as is.
- o Mapping to the PCEP MIB Module should be clear.
- o The data model should allow for static configurations of peers.
- o The data model should include read-only counters in order to gather statistics for sent and received PCEP messages, received messages with errors, and messages that could not be sent due to errors.
- o It should be fairly straightforward to augment the base data model for advanced PCE features.





## 5. The Design of PCEP Data Model

The module, "ietf-pcep", defines the basic components of a PCE speaker.

```

module: ietf-pcep
  +--rw pcep!
  |   +--rw entity
  |   |   +--rw addr inet:ip-address
  |   |   +--rw enabled? boolean
  |   |   +--rw role pcep-role
  |   |   +--rw description? string
  |   |   +--rw speaker-entity-id? string {stateful-sync-opt}?
  |   |   +--rw domain
  |   |   |   +--rw domain* [domain-type domain]
  |   |   |   |   +--rw domain-type domain-type
  |   |   |   |   +--rw domain domain
  |   |   +--rw capability
  |   |   |   +--rw gmpls? boolean {gmpls}?
  |   |   |   +--rw bi-dir? boolean
  |   |   |   +--rw diverse? boolean
  |   |   |   +--rw load-balance? boolean
  |   |   |   +--rw synchronize? boolean {svec}?
  |   |   |   +--rw objective-function? boolean {objective-function}?
  |   |   |   +--rw add-path-constraint? boolean
  |   |   |   +--rw prioritization? boolean
  |   |   |   +--rw multi-request? boolean
  |   |   |   +--rw gco? boolean {gco}?
  |   |   |   +--rw p2mp? boolean {p2mp}?
  |   |   |   +--rw stateful {stateful}?
  |   |   |   |   +--rw enabled? boolean
  |   |   |   |   +--rw active? boolean
  |   |   |   |   +--rw pce-initiated? boolean {pce-initiated}?
  |   |   |   |   +--rw include-db-ver? boolean {stateful-sync-opt}?
  |   |   |   |   +--rw trigger-resync? boolean {stateful-sync-opt}?
  |   |   |   |   +--rw trigger-initial-sync? boolean {stateful-sync-opt}?
  |   |   |   |   +--rw incremental-sync? boolean {stateful-sync-opt}?
  |   |   |   +--rw sr {sr}?
  |   |   |   |   +--rw enabled? boolean
  |   |   +--rw pce-info
  |   |   |   +--rw scope
  |   |   |   |   +--rw intra-area-scope? boolean
  |   |   |   |   +--rw intra-area-pref? uint8
  |   |   |   |   +--rw inter-area-scope? boolean
  |   |   |   |   +--rw inter-area-scope-default? boolean
  |   |   |   |   +--rw inter-area-pref? uint8
  |   |   |   |   +--rw inter-as-scope? boolean
  |   |   |   |   +--rw inter-as-scope-default? boolean

```



```

|   |   |   +---rw inter-as-pref?                uint8
|   |   |   +---rw inter-layer-scope?            boolean
|   |   |   +---rw inter-layer-pref?            uint8
|   |   +---rw neigh-domains
|   |   |   +---rw domain* [domain-type domain]
|   |   |       +---rw domain-type    domain-type
|   |   |       +---rw domain        domain
|   |   +---rw path-key {path-key}?
|   |       +---rw enabled?            boolean
|   |       +---rw discard-timer?    uint32
|   |       +---rw reuse-time?       uint32
|   |       +---rw pce-id?           inet:ip-address
| +---rw (auth-type-selection)?
|   |   +---:(auth-key-chain)
|   |   |   +---rw key-chain?                key-chain:key-chain-ref
|   |   +---:(auth-key)
|   |   |   +---rw key?                      string
|   |   |   +---rw crypto-algorithm
|   |   |       +---rw (algorithm)?
|   |   |           +---:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
|   |   |               |   +---rw hmac-sha1-12?                empty
|   |   |           +---:(aes-cmac-prf-128) {aes-cmac-prf-128}?
|   |   |               |   +---rw aes-cmac-prf-128?            empty
|   |   |           +---:(md5)
|   |   |               |   +---rw md5?                          empty
|   |   |           +---:(sha-1)
|   |   |               |   +---rw sha-1?                        empty
|   |   |           +---:(hmac-sha-1)
|   |   |               |   +---rw hmac-sha-1?                  empty
|   |   |           +---:(hmac-sha-256)
|   |   |               |   +---rw hmac-sha-256?                empty
|   |   |           +---:(hmac-sha-384)
|   |   |               |   +---rw hmac-sha-384?                empty
|   |   |           +---:(hmac-sha-512)
|   |   |               |   +---rw hmac-sha-512?                empty
|   |   |           +---:(clear-text) {clear-text}?
|   |   |               |   +---rw clear-text?                  empty
|   |   |           +---:(replay-protection-only) {replay-protection-only}?
|   |   |               +---rw replay-protection-only?    empty
|   |   +---:(auth-tls) {tls}?
|   |       +---rw tls
| +---rw connect-timer?                uint32
| +---rw connect-max-retry?            uint32
| +---rw init-backoff-timer?           uint32
| +---rw max-backoff-timer?            uint32
| +---rw open-wait-timer?              uint32
| +---rw keep-wait-timer?              uint32
| +---rw keep-alive-timer?             uint32

```



```
|   +--rw dead-timer?                uint32
|   +--rw allow-negotiation?         boolean
|   +--rw max-keep-alive-timer?      uint32
|   +--rw max-dead-timer?            uint32
|   +--rw min-keep-alive-timer?      uint32
|   +--rw min-dead-timer?            uint32
|   +--rw sync-timer?                uint32 {svec}?
|   +--rw request-timer?             uint32
|   +--rw max-sessions?              uint32
|   +--rw max-unknown-reqs?          uint32
|   +--rw max-unknown-msgs?          uint32
|   +--rw pcep-notification-max-rate uint32
|   +--rw stateful-parameter {stateful}?
|   |   +--rw state-timeout?          uint32
|   |   +--rw redelegation-timeout?   uint32
|   |   +--rw rpt-non-pcep-lsp?       boolean
|   +--rw of-list {objective-function}?
|   |   +--rw objective-function* [of]
|   |   |   +--rw of                 objective-function
|   +--rw peers
|   |   +--rw peer* [addr]
|   |   |   +--rw addr                inet:ip-address
|   |   |   +--rw description?        string
|   |   |   +--rw domain
|   |   |   |   +--rw domain* [domain-type domain]
|   |   |   |   |   +--rw domain-type domain-type
|   |   |   |   |   +--rw domain      domain
|   |   +--rw capability
|   |   |   +--rw gmpls?               boolean {gmpls}?
|   |   |   +--rw bi-dir?              boolean
|   |   |   +--rw diverse?             boolean
|   |   |   +--rw load-balance?         boolean
|   |   |   +--rw synchronize?         boolean {svec}?
|   |   |   +--rw objective-function?  boolean {objective-function}?
|   |   |   +--rw add-path-constraint? boolean
|   |   |   +--rw prioritization?       boolean
|   |   |   +--rw multi-request?        boolean
|   |   |   +--rw gco?                 boolean {gco}?
|   |   |   +--rw p2mp?                boolean {p2mp}?
|   |   |   +--rw stateful {stateful}?
|   |   |   |   +--rw enabled?          boolean
|   |   |   |   +--rw active?           boolean
|   |   |   |   +--rw pce-initiated?    boolean {pce-initiated}?
|   |   |   |   +--rw include-db-ver?   boolean {stateful-sync-opt}?
|   |   |   |   +--rw trigger-resync?   boolean {stateful-sync-opt}?
|   |   |   |   +--rw trigger-initial-sync? boolean {stateful-sync-opt}?
|   |   |   |   +--rw incremental-sync? boolean {stateful-sync-opt}?
|   |   +--rw sr {sr}?
|
```



```

|         |         +--rw enabled?    boolean
|
| +--rw scope
|         |         +--rw intra-area-scope?          boolean
|         |         +--rw intra-area-pref?           uint8
|         |         +--rw inter-area-scope?          boolean
|         |         +--rw inter-area-scope-default?  boolean
|         |         +--rw inter-area-pref?           uint8
|         |         +--rw inter-as-scope?            boolean
|         |         +--rw inter-as-scope-default?    boolean
|         |         +--rw inter-as-pref?             uint8
|         |         +--rw inter-layer-scope?         boolean
|         |         +--rw inter-layer-pref?          uint8
|
| +--rw neigh-domains
|         |         +--rw domain* [domain-type domain]
|         |         |         +--rw domain-type    domain-type
|         |         |         +--rw domain         domain
|
| +--rw delegation-pref?    uint8 {stateful}?
| +--rw (auth-type-selection)?
|         |         +--:(auth-key-chain)
|         |         |         +--rw key-chain?          key-chain:key-chain-ref
|         |         +--:(auth-key)
|         |         |         +--rw key?                string
|         |         |         +--rw crypto-algorithm
|         |         |         |         +--rw (algorithm)?
|         |         |         |         |         +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
|         |         |         |         |         |         +--rw hmac-sha1-12?          empty
|         |         |         |         |         +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
|         |         |         |         |         |         +--rw aes-cmac-prf-128?      empty
|         |         |         |         +--:(md5)
|         |         |         |         |         +--rw md5?                          empty
|         |         |         |         +--:(sha-1)
|         |         |         |         |         +--rw sha-1?                      empty
|         |         |         |         +--:(hmac-sha-1)
|         |         |         |         |         +--rw hmac-sha-1?                empty
|         |         |         |         +--:(hmac-sha-256)
|         |         |         |         |         +--rw hmac-sha-256?              empty
|         |         |         |         +--:(hmac-sha-384)
|         |         |         |         |         +--rw hmac-sha-384?              empty
|         |         |         |         +--:(hmac-sha-512)
|         |         |         |         |         +--rw hmac-sha-512?              empty
|         |         |         |         +--:(clear-text) {clear-text}?
|         |         |         |         |         +--rw clear-text?                empty
|         |         |         +--:(replay-protection-only) {replay-protection-
only}}?
|         |         |         |         +--rw replay-protection-only?    empty
|         |         +--:(auth-tls) {tls}?
|         |         +--rw tls
|
| +--rw of-list {objective-function}?

```



|                   +--rw objective-function\* [of]

```

|               +--rw of      objective-function
+--ro pcep-state
  +--ro entity
    +--ro addr?                inet:ip-address
    +--ro index?              uint32
    +--ro admin-status?       pcep-admin-status
    +--ro oper-status?        pcep-admin-status
    +--ro role?               pcep-role
    +--ro description?         string
    +--ro speaker-entity-id?   string {stateful-sync-opt}?
    +--ro domain
      | +--ro domain* [domain-type domain]
      |   +--ro domain-type    domain-type
      |   +--ro domain         domain
    +--ro capability
      | +--ro gmpls?           boolean {gmpls}?
      | +--ro bi-dir?          boolean
      | +--ro diverse?         boolean
      | +--ro load-balance?    boolean
      | +--ro synchronize?     boolean {svec}?
      | +--ro objective-function? boolean {objective-function}?
      | +--ro add-path-constraint? boolean
      | +--ro prioritization?  boolean
      | +--ro multi-request?   boolean
      | +--ro gco?             boolean {gco}?
      | +--ro p2mp?            boolean {p2mp}?
      | +--ro stateful {stateful}?
      | | +--ro enabled?       boolean
      | | +--ro active?        boolean
      | | +--ro pce-initiated? boolean {pce-initiated}?
      | | +--ro include-db-ver? boolean {stateful-sync-opt}?
      | | +--ro trigger-resync? boolean {stateful-sync-opt}?
      | | +--ro trigger-initial-sync? boolean {stateful-sync-opt}?
      | | +--ro incremental-sync? boolean {stateful-sync-opt}?
      | +--ro sr {sr}?
      |   +--ro enabled?       boolean
    +--ro pce-info
      | +--ro scope
      | | +--ro intra-area-scope?    boolean
      | | +--ro intra-area-pref?     uint8
      | | +--ro inter-area-scope?    boolean
      | | +--ro inter-area-scope-default? boolean
      | | +--ro inter-area-pref?     uint8
      | | +--ro inter-as-scope?      boolean
      | | +--ro inter-as-scope-default? boolean
      | | +--ro inter-as-pref?       uint8
      | | +--ro inter-layer-scope?   boolean
      | | +--ro inter-layer-pref?    uint8

```



```
| +---ro neigh-domains
| | +---ro domain* [domain-type domain]
| |   +---ro domain-type    domain-type
| |   +---ro domain         domain
| +---ro path-key {path-key}?
|   +---ro enabled?         boolean
|   +---ro discard-timer?   uint32
|   +---ro reuse-time?      uint32
|   +---ro pce-id?          inet:ip-address
+---ro (auth-type-selection)?
| +---:(auth-key-chain)
| | +---ro key-chain?        key-chain:key-chain-ref
| +---:(auth-key)
| | +---ro key?              string
| | +---ro crypto-algorithm
| |   +---ro (algorithm)?
| |     +---:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
| |       | +---ro hmac-sha1-12?          empty
| |       +---:(aes-cmac-prf-128) {aes-cmac-prf-128}?
| |         | +---ro aes-cmac-prf-128?      empty
| |         +---:(md5)
| |           | +---ro md5?                 empty
| |           +---:(sha-1)
| |             | +---ro sha-1?              empty
| |             +---:(hmac-sha-1)
| |               | +---ro hmac-sha-1?        empty
| |               +---:(hmac-sha-256)
| |                 | +---ro hmac-sha-256?      empty
| |                 +---:(hmac-sha-384)
| |                   | +---ro hmac-sha-384?      empty
| |                   +---:(hmac-sha-512)
| |                     | +---ro hmac-sha-512?      empty
| |                     +---:(clear-text) {clear-text}?
| |                       | +---ro clear-text?      empty
| |                       +---:(replay-protection-only) {replay-protection-only}?
| |                         +---ro replay-protection-only? empty
| +---:(auth-tls) {tls}?
|   +---ro tls
+---ro connect-timer?        uint32
+---ro connect-max-retry?    uint32
+---ro init-backoff-timer?   uint32
+---ro max-backoff-timer?    uint32
+---ro open-wait-timer?      uint32
+---ro keep-wait-timer?      uint32
+---ro keep-alive-timer?     uint32
+---ro dead-timer?           uint32
+---ro allow-negotiation?    boolean
+---ro max-keep-alive-timer? uint32
```



```

+--ro max-dead-timer?          uint32
+--ro min-keep-alive-timer?    uint32
+--ro min-dead-timer?          uint32
+--ro sync-timer?              uint32 {svec}?
+--ro request-timer?           uint32
+--ro max-sessions?            uint32
+--ro max-unknown-reqs?        uint32
+--ro max-unknown-msgs?        uint32
+--ro stateful-parameter {stateful}?
| +--ro state-timeout?          uint32
| +--ro redelegation-timeout?   uint32
| +--ro rpt-non-pcep-lsp?       boolean
+--ro lsp-db {stateful}?
| +--ro db-ver?                 uint64 {stateful-sync-opt}?
| +--ro association-list* [id source global-source extended-id]
| | +--ro type?                 assoc-type
| | +--ro id                    uint16
| | +--ro source                inet:ip-address
| | +--ro global-source         uint32
| | +--ro extended-id          string
| | +--ro lsp* [plsp-id pcc-id]
| |   +--ro plsp-id             -> /pcep-state/entity/lsp-db/lsp/plsp-id
| |   +--ro pcc-id              -> /pcep-state/entity/lsp-db/lsp/pcc-id
| +--ro lsp* [plsp-id pcc-id]
|   +--ro plsp-id               uint32
|   +--ro pcc-id                inet:ip-address
|   +--ro lsp-ref
|   | +--ro source?              -> /te:te/lsp-state/lsp/source
|   | +--ro destination?         -> /te:te/lsp-state/lsp/
destination
|   | +--ro tunnel-id?           -> /te:te/lsp-state/lsp/tunnel-
id
|   | +--ro lsp-id?              -> /te:te/lsp-state/lsp/lsp-id
|   | +--ro extended-tunnel-id?  -> /te:te/lsp-state/lsp/
extended-tunnel-id
|   | +--ro type?                -> /te:te/lsp-state/lsp/type
|   +--ro admin-state?           boolean
|   +--ro operational-state?     operational-state
|   +--ro delegated
|   | +--ro enabled?             boolean
|   | +--ro pce?                 -> /pcep-state/entity/peers/peer/addr
|   | +--ro srp-id?             uint32
|   +--ro initiation {pce-initiated}?
|   | +--ro enabled?             boolean
|   | +--ro pce?                 -> /pcep-state/entity/peers/peer/addr
|   +--ro symbolic-path-name?   string
|   +--ro last-error?            lsp-error
|   +--ro pst?                  pst

```

```

    |      +--ro association-list* [id source global-source extended-id]
    |      +--ro id                  -> /pcep-state/entity/lsp-db/
association-list/id
    |      +--ro source              -> /pcep-state/entity/lsp-db/
association-list/source
```

```

    |      +--ro global-source    -> /pcep-state/entity/lsp-db/
association-list/global-source
    |      +--ro extended-id      -> /pcep-state/entity/lsp-db/
association-list/extended-id
+--ro path-keys {path-key}?
| +--ro path-keys* [path-key]
|   +--ro path-key          uint16
|   +--ro cps
|   | +--ro explicit-route-objects* [index]
|   | | +--ro index          uint8
|   | | +--ro explicit-route-usage? identityref
|   | | +--ro (type)?
|   | |   +--:(ipv4-address)
|   | |   | +--ro v4-address?      inet:ipv4-address
|   | |   | +--ro v4-prefix-length? uint8
|   | |   | +--ro v4-loose?        boolean
|   | |   +--:(ipv6-address)
|   | |   | +--ro v6-address?      inet:ipv6-address
|   | |   | +--ro v6-prefix-length? uint8
|   | |   | +--ro v6-loose?        boolean
|   | |   +--:(as-number)
|   | |   | +--ro as-number?        uint16
|   | |   +--:(unnumbered-link)
|   | |   | +--ro router-id?        inet:ip-address
|   | |   | +--ro interface-id?     uint32
|   | |   +--:(label)
|   | |   +--ro value?              uint32
|   +--ro pcc-original?    -> /pcep-state/entity/peers/peer/addr
|   +--ro req-id?          uint32
|   +--ro retrieved?        boolean
|   +--ro pcc-retrieved?    -> /pcep-state/entity/peers/peer/addr
|   +--ro creation-time?    yang:timestamp
|   +--ro discard-time?     uint32
|   +--ro reuse-time?       uint32
+--ro of-list {objective-function}?
| +--ro objective-function* [of]
|   +--ro of      objective-function
+--ro peers
  +--ro peer* [addr]
    +--ro addr          inet:ip-address
    +--ro role?          pcep-role
    +--ro domain
    | +--ro domain* [domain-type domain]
    |   +--ro domain-type domain-type
    |   +--ro domain      domain
    +--ro capability
    | +--ro gmpls?          boolean {gmpls}?
    | +--ro bi-dir?         boolean

```



	+-ro diverse?	boolean
	+-ro load-balance?	boolean

```

| +--ro synchronize?          boolean {svec}?
| +--ro objective-function?   boolean {objective-function}?
| +--ro add-path-constraint?   boolean
| +--ro prioritization?        boolean
| +--ro multi-request?         boolean
| +--ro gco?                   boolean {gco}?
| +--ro p2mp?                  boolean {p2mp}?
| +--ro stateful {stateful}?
| | +--ro enabled?            boolean
| | +--ro active?             boolean
| | +--ro pce-initiated?      boolean {pce-initiated}?
| | +--ro include-db-ver?     boolean {stateful-sync-opt}?
| | +--ro trigger-resync?     boolean {stateful-sync-opt}?
| | +--ro trigger-initial-sync? boolean {stateful-sync-opt}?
| | +--ro incremental-sync?   boolean {stateful-sync-opt}?
| +--ro sr {sr}?
|   +--ro enabled?            boolean
+--ro pce-info
| +--ro scope
| | +--ro intra-area-scope?    boolean
| | +--ro intra-area-pref?     uint8
| | +--ro inter-area-scope?    boolean
| | +--ro inter-area-scope-default? boolean
| | +--ro inter-area-pref?     uint8
| | +--ro inter-as-scope?      boolean
| | +--ro inter-as-scope-default? boolean
| | +--ro inter-as-pref?       uint8
| | +--ro inter-layer-scope?   boolean
| | +--ro inter-layer-pref?    uint8
| +--ro neigh-domains
|   +--ro domain* [domain-type domain]
|     +--ro domain-type        domain-type
|     +--ro domain              domain
+--ro delegation-pref?         uint8 {stateful}?
+--ro (auth-type-selection)?
| +--:(auth-key-chain)
| | +--ro key-chain?           key-chain:key-chain-ref
| +--:(auth-key)
| | +--ro key?                 string
| | +--ro crypto-algorithm
| |   +--ro (algorithm)?
| |     +--:(hmac-sha-1-12) {crypto-hmac-sha-1-12}?
| |       | +--ro hmac-sha1-12?          empty
| |     +--:(aes-cmac-prf-128) {aes-cmac-prf-128}?
| |       | +--ro aes-cmac-prf-128?      empty
| |     +--:(md5)
| |       | +--ro md5?                   empty
| |     +--:(sha-1)

```



```

| | | +--ro sha-1? empty
| | | +--:(hmac-sha-1)
| | | +--ro hmac-sha-1? empty
| | | +--:(hmac-sha-256)
| | | +--ro hmac-sha-256? empty
| | | +--:(hmac-sha-384)
| | | +--ro hmac-sha-384? empty
| | | +--:(hmac-sha-512)
| | | +--ro hmac-sha-512? empty
| | | +--:(clear-text) {clear-text}?
| | | +--ro clear-text? empty
| | | +--:(replay-protection-only) {replay-protection-
only}?
| | | +--ro replay-protection-only? empty
| | +--:(auth-tls) {tls}?
| | +--ro tls
+--ro of-list {objective-function}?
| +--ro objective-function* [of]
| +--ro of objective-function
+--ro discontinuity-time? yang:timestamp
+--ro initiate-session? boolean
+--ro session-exists? boolean
+--ro num-sess-setup-ok? yang:counter32
+--ro num-sess-setup-fail? yang:counter32
+--ro session-up-time? yang:timestamp
+--ro session-fail-time? yang:timestamp
+--ro session-fail-up-time? yang:timestamp
+--ro pcep-stats
| +--ro avg-rsp-time? uint32
| +--ro lwm-rsp-time? uint32
| +--ro hwm-rsp-time? uint32
| +--ro num-pcreq-sent? yang:counter32
| +--ro num-pcreq-rcvd? yang:counter32
| +--ro num-pcrep-sent? yang:counter32
| +--ro num-pcrep-rcvd? yang:counter32
| +--ro num-pcerr-sent? yang:counter32
| +--ro num-pcerr-rcvd? yang:counter32
| +--ro num-pcntf-sent? yang:counter32
| +--ro num-pcntf-rcvd? yang:counter32
| +--ro num-keepalive-sent? yang:counter32
| +--ro num-keepalive-rcvd? yang:counter32
| +--ro num-unknown-rcvd? yang:counter32
| +--ro num-corrupt-rcvd? yang:counter32
| +--ro num-req-sent? yang:counter32
| +--ro num-req-sent-pend-rep? yang:counter32
| +--ro num-req-sent-ero-rcvd? yang:counter32
| +--ro num-req-sent-nopath-rcvd? yang:counter32
| +--ro num-req-sent-cancel-rcvd? yang:counter32

```

| +--ro num-req-sent-error-rcvd? yang:counter32

```

| +--ro num-req-sent-timeout?      yang:counter32
| +--ro num-req-sent-cancel-sent?  yang:counter32
| +--ro num-req-rcvd?              yang:counter32
| +--ro num-req-rcvd-pend-rep?     yang:counter32
| +--ro num-req-rcvd-ero-sent?     yang:counter32
| +--ro num-req-rcvd-nopath-sent?  yang:counter32
| +--ro num-req-rcvd-cancel-sent?  yang:counter32
| +--ro num-req-rcvd-error-sent?   yang:counter32
| +--ro num-req-rcvd-cancel-rcvd?  yang:counter32
| +--ro num-rep-rcvd-unknown?      yang:counter32
| +--ro num-req-rcvd-unknown?      yang:counter32
| +--ro svec {svec}?
| | +--ro num-svec-sent?           yang:counter32
| | +--ro num-svec-req-sent?       yang:counter32
| | +--ro num-svec-rcvd?           yang:counter32
| | +--ro num-svec-req-rcvd?       yang:counter32
| +--ro stateful {stateful}?
| | +--ro num-pcrpt-sent?           yang:counter32
| | +--ro num-pcrpt-rcvd?           yang:counter32
| | +--ro num-pcupd-sent?           yang:counter32
| | +--ro num-pcupd-rcvd?           yang:counter32
| | +--ro num-rpt-sent?             yang:counter32
| | +--ro num-rpt-rcvd?             yang:counter32
| | +--ro num-rpt-rcvd-error-sent?  yang:counter32
| | +--ro num-upd-sent?             yang:counter32
| | +--ro num-upd-rcvd?             yang:counter32
| | +--ro num-upd-rcvd-unknown?     yang:counter32
| | +--ro num-upd-rcvd-undelegated? yang:counter32
| | +--ro num-upd-rcvd-error-sent?  yang:counter32
| | +--ro initiation {pce-initiated}?
| | | +--ro num-pcinitiate-sent?    yang:counter32
| | | +--ro num-pcinitiate-rcvd?    yang:counter32
| | | +--ro num-initiate-sent?      yang:counter32
| | | +--ro num-initiate-rcvd?      yang:counter32
| | | +--ro num-initiate-rcvd-error-sent? yang:counter32
| +--ro path-key {path-key}?
| | +--ro num-unknown-path-key?     yang:counter32
| | +--ro num-exp-path-key?         yang:counter32
| | +--ro num-dup-path-key?         yang:counter32
| | +--ro num-path-key-no-attempt?  yang:counter32
| +--ro num-req-sent-closed?        yang:counter32
| +--ro num-req-rcvd-closed?        yang:counter32
+--ro sessions
  +--ro session* [initiator]
    +--ro initiator                  pcep-initiator
    +--ro state-last-change?         yang:timestamp
    +--ro state?                     pcep-sess-state
    +--ro session-creation?          yang:timestamp

```



```

+--ro connect-retry?          yang:counter32
+--ro local-id?               uint32
+--ro remote-id?              uint32
+--ro keepalive-timer?        uint32
+--ro peer-keepalive-timer?   uint32
+--ro dead-timer?             uint32
+--ro peer-dead-timer?        uint32
+--ro ka-hold-time-rem?        uint32
+--ro overloaded?              boolean
+--ro overload-time?           uint32
+--ro peer-overloaded?         boolean
+--ro peer-overload-time?      uint32
+--ro lspdb-sync?              sync-state {stateful}?
+--ro recv-db-ver?             uint64 {stateful,stateful-
sync-opt}?

+--ro of-list {objective-function}?
| +--ro objective-function* [of]
|   +--ro of      objective-function
+--ro speaker-entity-id?       string {stateful-sync-opt}?
+--ro discontinuity-time?       yang:timestamp
+--ro pcep-stats
  +--ro avg-rsp-time?           uint32
  +--ro lwm-rsp-time?           uint32
  +--ro hwm-rsp-time?           uint32
  +--ro num-pcreq-sent?          yang:counter32
  +--ro num-pcreq-rcvd?          yang:counter32
  +--ro num-pcrep-sent?          yang:counter32
  +--ro num-pcrep-rcvd?          yang:counter32
  +--ro num-pcerr-sent?          yang:counter32
  +--ro num-pcerr-rcvd?          yang:counter32
  +--ro num-pcntf-sent?          yang:counter32
  +--ro num-pcntf-rcvd?          yang:counter32
  +--ro num-keepalive-sent?       yang:counter32
  +--ro num-keepalive-rcvd?       yang:counter32
  +--ro num-unknown-rcvd?         yang:counter32
  +--ro num-corrupt-rcvd?         yang:counter32
  +--ro num-req-sent?             yang:counter32
  +--ro num-req-sent-pend-rep?     yang:counter32
  +--ro num-req-sent-ero-rcvd?     yang:counter32
  +--ro num-req-sent-nopath-rcvd?  yang:counter32
  +--ro num-req-sent-cancel-rcvd?  yang:counter32
  +--ro num-req-sent-error-rcvd?   yang:counter32
  +--ro num-req-sent-timeout?      yang:counter32
  +--ro num-req-sent-cancel-sent?  yang:counter32
  +--ro num-req-rcvd?             yang:counter32
  +--ro num-req-rcvd-pend-rep?     yang:counter32
  +--ro num-req-rcvd-ero-sent?     yang:counter32
  +--ro num-req-rcvd-nopath-sent?  yang:counter32

```



`+-ro num-req-rcvd-cancel-sent? yang:counter32`

```

    +--ro num-req-rcvd-error-sent?      yang:counter32
    +--ro num-req-rcvd-cancel-rcvd?     yang:counter32
    +--ro num-rep-rcvd-unknown?         yang:counter32
    +--ro num-req-rcvd-unknown?         yang:counter32
    +--ro svec {svec}?
      | +--ro num-svec-sent?             yang:counter32
      | +--ro num-svec-req-sent?        yang:counter32
      | +--ro num-svec-rcvd?            yang:counter32
      | +--ro num-svec-req-rcvd?        yang:counter32
    +--ro stateful {stateful}?
      | +--ro num-pcrpt-sent?            yang:counter32
      | +--ro num-pcrpt-rcvd?            yang:counter32
      | +--ro num-pcupd-sent?            yang:counter32
      | +--ro num-pcupd-rcvd?            yang:counter32
      | +--ro num-rpt-sent?              yang:counter32
      | +--ro num-rpt-rcvd?              yang:counter32
      | +--ro num-rpt-rcvd-error-sent?   yang:counter32
      | +--ro num-upd-sent?              yang:counter32
      | +--ro num-upd-rcvd?              yang:counter32
      | +--ro num-upd-rcvd-unknown?      yang:counter32
      | +--ro num-upd-rcvd-undelegated?  yang:counter32
      | +--ro num-upd-rcvd-error-sent?   yang:counter32
      | +--ro initiation {pce-initiated}?
      |   +--ro num-pcinitiate-sent?
yang:counter32
      |   +--ro num-pcinitiate-rcvd?
yang:counter32
      |   +--ro num-initiate-sent?
yang:counter32
      |   +--ro num-initiate-rcvd?
yang:counter32
      |   +--ro num-initiate-rcvd-error-sent?
yang:counter32
    +--ro path-key {path-key}?
      +--ro num-unknown-path-key?       yang:counter32
      +--ro num-exp-path-key?           yang:counter32
      +--ro num-dup-path-key?           yang:counter32
      +--ro num-path-key-no-attempt?    yang:counter32

notifications:
  +---n pcep-session-up
    | +--ro peer-addr?                  -> /pcep-state/entity/peers/peer/addr
    | +--ro session-initiator?          -> /pcep-state/entity/peers/peer/sessions/
session/initiator
    | +--ro state-last-change?          yang:timestamp
    | +--ro state?                      pcep-sess-state
  +---n pcep-session-down
    | +--ro peer-addr?                  -> /pcep-state/entity/peers/peer/addr
    | +--ro session-initiator?          pcep-initiator

```

```
|  +-ro state-last-change?  yang:timestamp
|  +-ro state?               pcep-sess-state
+---n pcep-session-local-overload
|  +-ro peer-addr?          -> /pcep-state/entity/peers/peer/addr
|  +-ro session-initiator?  -> /pcep-state/entity/peers/peer/sessions/
session/initiator
|  +-ro overloaded?         boolean
```

```
|  +--ro overload-time?          uint32
+---n pcep-session-local-overload-clear
|  +--ro peer-addr?              -> /pcep-state/entity/peers/peer/addr
|  +--ro overloaded?             boolean
+---n pcep-session-peer-overload
|  +--ro peer-addr?              -> /pcep-state/entity/peers/peer/addr
|  +--ro session-initiator?      -> /pcep-state/entity/peers/peer/sessions/
session/initiator
|  +--ro peer-overloaded?         boolean
|  +--ro peer-overload-time?     uint32
+---n pcep-session-peer-overload-clear
    +--ro peer-addr?              -> /pcep-state/entity/peers/peer/addr
    +--ro peer-overloaded?        boolean
```

### 5.1. The Entity

The PCEP yang module may contain status information for the local PCEP entity.

The entity has an IP address (using ietf-inet-types [[RFC6991](#)]) and a "role" leaf (the local entity PCEP role) as mandatory.

Note that, the PCEP MIB module [[RFC7420](#)] uses an entity list and a system generated entity index as a primary index to the read only entity table. If the device implements the PCEP MIB, the "index" leaf MUST contain the value of the corresponding pcePcepEntityIndex and only one entity is assumed.

### 5.2. The Peer Lists

The peer list contains peer(s) that the local PCEP entity knows about. A PCEP speaker is identified by its IP address. If there is a PCEP speaker in the network that uses multiple IP addresses then it looks like multiple distinct peers to the other PCEP speakers in the network.

Since PCEP sessions can be ephemeral, the peer list tracks a peer even when no PCEP session currently exists to that peer. The statistics contained are an aggregate of the statistics for all successive sessions to that peer.

To limit the quantity of information that is stored, an implementation MAY choose to discard this information if and only if no PCEP session exists to the corresponding peer.

The data model for PCEP peer presented in this document uses a flat list of peers. Each peer in the list is identified by its IP address (addr-type, addr).



There is one list for static peer configuration ("/pcep/entity/peers"), and a separate list for the operational state of all peers (i.e. static as well as discovered)("/pcep-state/entity/peers"). The former is used to enable remote PCE configuration at PCC (or PCE) while the latter has the operational state of these peers as well as the remote PCE peer which were discovered and PCC peers that have initiated session.

### **5.3. The Session Lists**

The session list contains PCEP session that the PCEP entity (PCE or PCC) is currently participating in. The statistics in session are semantically different from those in peer since the former applies to the current session only, whereas the latter is the aggregate for all sessions that have existed to that peer.

Although [[RFC5440](#)] forbids more than one active PCEP session between a given pair of PCEP entities at any given time, there is a window during session establishment where two sessions may exist for a given pair, one representing a session initiated by the local PCEP entity and the other representing a session initiated by the peer. If either of these sessions reaches active state first, then the other is discarded.

The data model for PCEP session presented in this document uses a flat list of sessions. Each session in the list is identified by its initiator. This index allows two sessions to exist transiently for a given peer, as discussed above.

There is only one list for the operational state of all sessions ("/pcep-state/entity/peers/peer/sessions/session").

### **5.4. Notifications**

This YANG model defines a list of notifications to inform client of important events detected during the protocol operation. The notifications defined cover the PCEP MIB notifications.

## **6. Advanced PCE Features**

This document contains a specification of the base PCEP YANG module, "ietf-pcep" which provides the basic PCEP [[RFC5440](#)] data model.

This document further handles advanced PCE features like -

- o Capability and Scope
- o Domain information (local/neighbour)



- o Path-Key
- o OF
- o GCO
- o P2MP
- o GMPLS
- o Inter-Layer
- o Stateful PCE
- o Segment Routing
- o Authentication including PCEPS (TLS)

[Editor's Note - Some of them would be added in a future revision.]

#### **6.1. Stateful PCE's LSP-DB**

In the operational state of PCEP which supports stateful PCE mode, the list of LSP state are maintained in LSP-DB. The key is the PLSP-ID and the PCC IP address.

The PCEP data model contains the operational state of LSPs (/pcep-state/entity/lsp-db/lsp/) with PCEP specific attributes. The generic TE attributes of the LSP are defined in [[I-D.ietf-teas-yang-te](#)]. A reference to LSP state in TE model is maintained.

### **7. Open Issues and Next Step**

This section is added so that open issues can be tracked. This section would be removed when the document is ready for publication.

#### **7.1. The PCE-Initiated LSP**

The TE Model at [[I-D.ietf-teas-yang-te](#)] should support creation of tunnels at the controller (PCE) and marking them as PCE-Initiated. The LSP-DB in the PCEP Yang (/pcep-state/entity/lsp-db/lsp/initiation) also marks the LSPs which are PCE-initiated.

#### **7.2. PCEP over TLS (PCEPS)**

A future version of this document would add TLS related configurations.





## 8. PCEP YANG Module

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

```
<CODE BEGINS> file "ietf-pcep@2016-10-27.yang"
module ietf-pcep {
  namespace "urn:ietf:params:xml:ns:yang:ietf-pcep";
  prefix pcep;

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

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

  import ietf-te {
    prefix "te";
  }

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

  import ietf-key-chain {
    prefix "key-chain";
  }

  organization
    "IETF PCE (Path Computation Element) Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/pce/>
    WG List:  <mailto:pce@ietf.org>
    WG Chair: JP Vasseur
               <mailto:jpv@cisco.com>
    WG Chair: Julien Meuric
               <mailto:julien.meuric@orange.com>
    WG Chair: Jonathan Hardwick
               <mailto:Jonathan.Hardwick@metaswitch.com>
    Editor:   Dhruv Dhody
               <mailto:dhruv.ietf@gmail.com>";
```



```
description
    "The YANG module defines a generic configuration and
    operational model for PCEP common across all of the
    vendor implementations.";

revision 2016-10-27 {
    description "Initial revision.";
    reference
        "RFC XXXX: A YANG Data Model for Path Computation
        Element Communications Protocol
        (PCEP)";
}

/*
 * Identities
 */

identity pcep {
    description "Identity for the PCEP protocol.";
}

/*
 * Typedefs
 */
typedef pcep-role {
    type enumeration {
        enum unknown {
            value "0";
            description
                "An unknown role";
        }
        enum pcc {
            value "1";
            description
                "The role of a Path Computation Client";
        }
        enum pce {
            value "2";
            description
                "The role of Path Computation Element";
        }
        enum pcc-and-pce {
            value "3";
            description
                "The role of both Path Computation Client and
                Path Computation Element";
        }
    }
}
```



```
    }

    description
      "The role of a PCEP speaker.
      Takes one of the following values
      - unknown(0): the role is not known.
      - pcc(1): the role is of a Path Computation
        Client (PCC).
      - pce(2): the role is of a Path Computation
        Server (PCE).
      - pccAndPce(3): the role is of both a PCC and
        a PCE.";

  }

  typedef pcep-admin-status {
    type enumeration {
      enum admin-status-up {
        value "1";
        description
          "Admin Status is Up";
      }
      enum admin-status-down {
        value "2";
        description
          "Admin Status is Down";
      }
    }
  }

  description
    "The Admin Status of the PCEP entity.
    Takes one of the following values
    - admin-status-up(1): Admin Status is Up.
    - admin-status-down(2): Admin Status is Down";
}

typedef pcep-oper-status {
  type enumeration {
    enum oper-status-up {
      value "1";
      description
        "The PCEP entity is active";
    }
    enum oper-status-down {
      value "2";
      description
        "The PCEP entity is inactive";
    }
  }
}
```



```
    enum oper-status-going-up {
        value "3";
        description
            "The PCEP entity is activating";
    }
    enum oper-status-going-down {
        value "4";
        description
            "The PCEP entity is deactivating";
    }
    enum oper-status-failed {
        value "5";
        description
            "The PCEP entity has failed and will recover
            when possible.";
    }
    enum oper-status-failed-perm {
        value "6";
        description
            "The PCEP entity has failed and will not recover
            without operator intervention";
    }
}
description
    "The operational status of the PCEP entity.
    Takes one of the following values
    - oper-status-up(1): Active
    - oper-status-down(2): Inactive
    - oper-status-going-up(3): Activating
    - oper-status-going-down(4): Deactivating
    - oper-status-failed(5): Failed
    - oper-status-failed-perm(6): Failed Permanantly";
}

typedef pcep-initiator {
    type enumeration {
        enum local {
            value "1";
            description
                "The local PCEP entity initiated the session";
        }

        enum remote {
            value "2";
            description
                "The remote PCEP peer initiated the session";
        }
    }
}
```





```
description
"The initiator of the session, that is, whether the TCP
connection was initiated by the local PCEP entity or
the remote peer.
Takes one of the following values
  - local(1): Initiated locally
  - remote(2): Initiated remotely";
}

typedef pcep-sess-state {
  type enumeration {
    enum tcp-pending {
      value "1";
      description
        "The tcp-pending state of PCEP session.";
    }

    enum open-wait {
      value "2";
      description
        "The open-wait state of PCEP session.";
    }

    enum keep-wait {
      value "3";
      description
        "The keep-wait state of PCEP session.";
    }

    enum session-up {
      value "4";
      description
        "The session-up state of PCEP session.";
    }
  }
}

description
"The current state of the session.
The set of possible states excludes the idle state
since entries do not exist in the idle state.
Takes one of the following values
  - tcp-pending(1): PCEP TCP Pending state
  - open-wait(2): PCEP Open Wait state
  - keep-wait(3): PCEP Keep Wait state
  - session-up(4): PCEP Session Up state";
}

typedef domain-type {
  type enumeration {
```



```
        enum ospf-area {
            value "1";
            description
                "The OSPF area.";
        }
        enum isis-area {
            value "2";
            description
                "The IS-IS area.";
        }
        enum as {
            value "3";
            description
                "The Autonomous System (AS).";
        }
    }
    description
        "The PCE Domain Type";
}

typedef domain-ospf-area {
    type union {
        type uint32;
        type yang:dotted-quad;
    }
    description
        "OSPF Area ID.";
}

typedef domain-isis-area {
    type string {
        pattern '[0-9A-Fa-f]{2}\.([0-9A-Fa-f]{4}\.){0,3}';
    }
    description
        "IS-IS Area ID.";
}

typedef domain-as {
    type uint32;
    description
        "Autonomous System number.";
}

typedef domain {
    type union {
        type domain-ospf-area;
        type domain-isis-area;
```



```
        type domain-as;
    }
    description
        "The Domain Information";
}

typedef operational-state {
    type enumeration {
        enum down {
            value "0";
            description
                "not active.";
        }
        enum up {
            value "1";
            description
                "signalled.";
        }
        enum active {
            value "2";
            description
                "up and carrying traffic.";
        }
        enum going-down {
            value "3";
            description
                "LSP is being torn down, resources are
                being released.";
        }
        enum going-up {
            value "4";
            description
                "LSP is being signalled.";
        }
    }
    description
        "The operational status of the LSP";
}

typedef lsp-error {
    type enumeration {
        enum no-error {
            value "0";
            description
                "No error, LSP is fine.";
        }
        enum unknown {
            value "1";
```



```
        description
            "Unknown reason.";
    }
    enum limit {
        value "2";
        description
            "Limit reached for PCE-controlled LSPs.";
    }
    enum pending {
        value "3";
        description
            "Too many pending LSP update requests.";
    }
    enum unacceptable {
        value "4";
        description
            "Unacceptable parameters.";
    }
    enum internal {
        value "5";
        description
            "Internal error.";
    }
    enum admin {
        value "6";
        description
            "LSP administratively brought down.";
    }
    enum preempted {
        value "7";
        description
            "LSP preempted.";
    }
    enum rsvp {
        value "8";
        description
            "RSVP signaling error.";
    }
}
description
    "The LSP Error Codes.";
}

typedef sync-state {
    type enumeration {
        enum pending {
            value "0";
            description
```





```
        "The state synchronization
        has not started.";
    }
    enum ongoing {
        value "1";
        description
            "The state synchronization
            is ongoing.";
    }
    enum finished {
        value "2";
        description
            "The state synchronization
            is finished.";
    }
}
description
    "The LSP-DB state synchronization operational status.";
}

typedef pst{
    type enumeration{
        enum rsvp-te{
            value "0";
            description
                "RSVP-TE signaling protocol";
        }
        enum sr{
            value "1";
            description
                "Segment Routing Traffic Engineering";
        }
    }
    description
        "The Path Setup Type";
}

typedef assoc-type{
    type enumeration{
        enum protection{
            value "1";
            description
                "Path Protection Association Type";
        }
    }
    description
        "The PCEP Association Type";
}
```



```
typedef objective-function{
  type enumeration{
    enum mcp{
      value "1";
      description
        "Minimum Cost Path (MCP)";
    }
    enum mlp{
      value "2";
      description
        "Minimum Load Path (MLP)";
    }
    enum mbp{
      value "3";
      description
        "Maximum residual Bandwidth Path (MBP)";
    }
    enum mbc{
      value "4";
      description
        "Minimize aggregate Bandwidth Consumption (MBC)";
    }
    enum mll{
      value "5";
      description
        "Minimize the Load of the most loaded Link (MLL)";
    }
    enum mcc{
      value "6";
      description
        "Minimize the Cumulative Cost of a set of paths
        (MCC)";
    }
    enum spt{
      value "7";
      description
        "Shortest Path Tree (SPT)";
    }
    enum mct{
      value "8";
      description
        "Minimum Cost Tree (MCT)";
    }
    enum mplp{
      value "9";
      description
        "Minimum Packet Loss Path (MPLP)";
    }
  }
}
```



```
        enum mup{
            value "10";
            description
                "Maximum Under-Utilized Path (MUP)";
        }
        enum mrup{
            value "11";
            description
                "Maximum Reserved Under-Utilized Path (MRUP)";
        }
    }
    description
        "The PCEP Objective functions";
}

/*
 * Features
 */

feature svec {
    description
        "Support synchronized path computation.";
}

feature gmpls {
    description
        "Support GMPLS.";
}

feature objective-function {
    description
        "Support OF as per RFC 5541.";
}

feature gco {
    description
        "Support GCO as per RFC 5557.";
}

feature path-key {
    description
        "Support path-key as per RFC 5520.";
}

feature p2mp {
    description
        "Support P2MP as per RFC 6006.";
}
```



```
feature stateful {
    description
        "Support stateful PCE.";
}

feature stateful-sync-opt {
    description
        "Support stateful sync optimization";
}

feature pce-initiated {
    description
        "Support PCE-Initiated LSP.";
}

feature tls {
    description
        "Support PCEP over TLS.";
}

feature sr {
    description
        "Support Segment Routing for PCE.";
}

/*
 * Groupings
 */

grouping pcep-entity-info{
    description
        "This grouping defines the attributes for PCEP entity.";
    leaf connect-timer {
        type uint32 {
            range "1..65535";
        }
        units "seconds";
        default 60;
        description
            "The time in seconds that the PCEP entity will wait
            to establish a TCP connection with a peer. If a
            TCP connection is not established within this time
            then PCEP aborts the session setup attempt.";
        reference
            "RFC 5440: Path Computation Element (PCE)
            Communication Protocol (PCEP)";
    }
}
```





```
}

leaf connect-max-retry {
  type uint32;
  default 5;
  description
    "The maximum number of times the system tries to
    establish a TCP connection to a peer before the
    session with the peer transitions to the idle
    state.";
  reference
    "RFC 5440: Path Computation Element (PCE)
    Communication Protocol (PCEP)";
}

leaf init-backoff-timer {
  type uint32 {
    range "1..65535";
  }
  units "seconds";
  description
    "The initial back-off time in seconds for retrying
    a failed session setup attempt to a peer.
    The back-off time increases for each failed
    session setup attempt, until a maximum back-off
    time is reached. The maximum back-off time is
    max-backoff-timer.";
}

leaf max-backoff-timer {
  type uint32;
  units "seconds";
  description
    "The maximum back-off time in seconds for retrying
    a failed session setup attempt to a peer.
    The back-off time increases for each failed session
    setup attempt, until this maximum value is reached.
    Session setup attempts then repeat periodically
    without any further increase in back-off time.";
}

leaf open-wait-timer {
  type uint32 {
    range "1..65535";
  }
  units "seconds";
  default 60;
  description
```



```
    "The time in seconds that the PCEP entity will wait
    to receive an Open message from a peer after the
    TCP connection has come up.
    If no Open message is received within this time then
    PCEP terminates the TCP connection and deletes the
    associated sessions.";
  reference
    "RFC 5440: Path Computation Element (PCE)
    Communication Protocol (PCEP)";
}

leaf keep-wait-timer {
  type uint32 {
    range "1..65535";
  }
  units "seconds";
  default 60;
  description
    "The time in seconds that the PCEP entity will wait
    to receive a Keepalive or PCErr message from a peer
    during session initialization after receiving an
    Open message. If no Keepalive or PCErr message is
    received within this time then PCEP terminates the
    TCP connection and deletes the associated
    sessions.";
  reference
    "RFC 5440: Path Computation Element (PCE)
    Communication Protocol (PCEP)";
}

leaf keep-alive-timer {
  type uint32 {
    range "0..255";
  }
  units "seconds";
  default 30;
  description
    "The keep alive transmission timer that this PCEP
    entity will propose in the initial OPEN message of
    each session it is involved in. This is the
    maximum time between two consecutive messages sent
    to a peer. Zero means that the PCEP entity prefers
    not to send Keepalives at all.
    Note that the actual Keepalive transmission
    intervals, in either direction of an active PCEP
    session, are determined by negotiation between the
    peers as specified by RFC 5440, and so may differ
    from this configured value.";
```



```
        reference
          "RFC 5440: Path Computation Element (PCE)
            Communication Protocol (PCEP)";
    }

    leaf dead-timer {
      type uint32 {
        range "0..255";
      }
      units "seconds";
      must "(. > ../keep-alive-timer)" {
        error-message "The dead timer must be "
          + "larger than the keep alive timer";
        description
          "This value MUST be greater than
            keep-alive-timer.";
      }
      default 120;
      description
        "The dead timer that this PCEP entity will propose
          in the initial OPEN message of each session it is
          involved in. This is the time after which a peer
          should declare a session down if it does not
          receive any PCEP messages. Zero suggests that the
          peer does not run a dead timer at all." ;
      reference
        "RFC 5440: Path Computation Element (PCE)
          Communication Protocol (PCEP)";
    }

    leaf allow-negotiation{
      type boolean;
      description
        "Whether the PCEP entity will permit negotiation of
          session parameters.";
    }

    leaf max-keep-alive-timer{
      type uint32 {
        range "0..255";
      }
      units "seconds";
      description
        "In PCEP session parameter negotiation in seconds,
          the maximum value that this PCEP entity will
          accept from a peer for the interval between
          Keepalive transmissions. Zero means that the PCEP
```



```
        entity will allow no Keepalive transmission at
        all." ;
    }

    leaf max-dead-timer{
        type uint32 {
            range "0..255";
        }
        units "seconds";
        description
            "In PCEP session parameter negotiation in seconds,
            the maximum value that this PCEP entity will accept
            from a peer for the Dead timer. Zero means that
            the PCEP entity will allow not running a Dead
            timer.";
    }

    leaf min-keep-alive-timer{
        type uint32 {
            range "0..255";
        }
        units "seconds";
        description
            "In PCEP session parameter negotiation in seconds,
            the minimum value that this PCEP entity will
            accept for the interval between Keepalive
            transmissions. Zero means that the PCEP entity
            insists on no Keepalive transmission at all.";
    }

    leaf min-dead-timer{
        type uint32 {
            range "0..255";
        }
        units "seconds";
        description
            "In PCEP session parameter negotiation in
            seconds, the minimum value that this PCEP entity
            will accept for the Dead timer. Zero means that
            the PCEP entity insists on not running a Dead
            timer.";
    }

    leaf sync-timer{
        if-feature svec;
        type uint32 {
            range "0..65535";
        }
    }
```





```
    units "seconds";
    default 60;
    description
        "The value of SyncTimer in seconds is used in the
        case of synchronized path computation request
        using the SVEC object. Consider the case where a
        PCReq message is received by a PCE that contains
        the SVEC object referring to M synchronized path
        computation requests. If after the expiration of
        the SyncTimer all the M path computation requests
        have not been, received a protocol error is
        triggered and the PCE MUST cancel the whole set
        of path computation requests.
        The aim of the SyncTimer is to avoid the storage
        of unused synchronized requests should one of
        them get lost for some reasons (for example, a
        misbehaving PCC).
        Zero means that the PCEP entity does not use the
        SyncTimer.";
    reference
        "RFC 5440: Path Computation Element (PCE)
        Communication Protocol (PCEP)";
}

leaf request-timer{
    type uint32 {
        range "1..65535";
    }
    units "seconds";
    description
        "The maximum time that the PCEP entity will wait
        for a response to a PCReq message.";
}

leaf max-sessions{
    type uint32;
    description
        "Maximum number of sessions involving this PCEP
        entity that can exist at any time.";
}

leaf max-unknown-reqs{
    type uint32;
    default 5;
    description
        "The maximum number of unrecognized requests and
        replies that any session on this PCEP entity is
```



```
    willing to accept per minute before terminating
    the session.
    A PCRep message contains an unrecognized reply
    if it contains an RP object whose request ID
    does not correspond to any in-progress request
    sent by this PCEP entity.
    A PCReq message contains an unrecognized request
    if it contains an RP object whose request ID is
    zero.";
  reference
    "RFC 5440: Path Computation Element (PCE)
      Communication Protocol (PCEP)";
}

leaf max-unknown-msgs{
  type uint32;
  default 5;
  description
    "The maximum number of unknown messages that any
    session on this PCEP entity is willing to accept
    per minute before terminating the session.";
  reference
    "RFC 5440: Path Computation Element (PCE)
      Communication Protocol (PCEP)";
}

} // pcep-entity-info

grouping pce-scope{
  description
    "This grouping defines PCE path computation scope
    information which maybe relevant to PCE selection.
    This information corresponds to PCE auto-discovery
    information.";
  reference
    "RFC 5088: OSPF Protocol Extensions for Path
      Computation Element (PCE)
      Discovery
    RFC 5089: IS-IS Protocol Extensions for Path
      Computation Element (PCE)
      Discovery";
  leaf intra-area-scope{
    type boolean;
    default true;
    description
      "PCE can compute intra-area paths.";
  }
  leaf intra-area-pref{
```



```
        type uint8{
            range "0..7";
        }
        description
            "The PCE's preference for intra-area TE LSP
            computation.";
    }
    leaf inter-area-scope{
        type boolean;
        default false;
        description
            "PCE can compute inter-area paths.";
    }
    leaf inter-area-scope-default{
        type boolean;
        default false;
        description
            "PCE can act as a default PCE for inter-area
            path computation.";
    }
    leaf inter-area-pref{
        type uint8{
            range "0..7";
        }
        description
            "The PCE's preference for inter-area TE LSP
            computation.";
    }
    leaf inter-as-scope{
        type boolean;
        default false;
        description
            "PCE can compute inter-AS paths.";
    }
    leaf inter-as-scope-default{
        type boolean;
        default false;
        description
            "PCE can act as a default PCE for inter-AS
            path computation.";
    }
    leaf inter-as-pref{
        type uint8{
            range "0..7";
        }
        description
            "The PCE's preference for inter-AS TE LSP
            computation.";
```



```
    }
    leaf inter-layer-scope{
        type boolean;
        default false;
        description
            "PCE can compute inter-layer paths.";
    }
    leaf inter-layer-pref{
        type uint8{
            range "0..7";
        }
        description
            "The PCE's preference for inter-layer TE LSP
            computation.";
    }
} //pce-scope

grouping domain{
    description
        "This grouping specifies a Domain where the
        PCEP speaker has topology visibility.";
    leaf domain-type{
        type domain-type;
        description
            "The domain type.";
    }
    leaf domain{
        type domain;
        description
            "The domain Information.";
    }
} //domain

grouping capability{
    description
        "This grouping specifies a capability
        information of local PCEP entity. This maybe
        relevant to PCE selection as well. This
        information corresponds to PCE auto-discovery
        information.";
    reference
        "RFC 5088: OSPF Protocol Extensions for Path
        Computation Element (PCE)
        Discovery
        RFC 5089: IS-IS Protocol Extensions for Path
        Computation Element (PCE)
        Discovery";
    leaf gmpls{
```





```
        if-feature gmpls;
        type boolean;
        description
            "Path computation with GMPLS link
            constraints.";
    }
    leaf bi-dir{
        type boolean;
        description
            "Bidirectional path computation.";
    }
    leaf diverse{
        type boolean;
        description
            "Diverse path computation.";
    }
    leaf load-balance{
        type boolean;
        description
            "Load-balanced path computation.";
    }
    leaf synchronize{
        if-feature svec;
        type boolean;
        description
            "Synchronized paths computation.";
    }
    leaf objective-function{
        if-feature objective-function;
        type boolean;
        description
            "Support for multiple objective functions.";
    }
    leaf add-path-constraint{
        type boolean;
        description
            "Support for additive path constraints (max
            hop count, etc.).";
    }
    leaf prioritization{
        type boolean;
        description
            "Support for request prioritization.";
    }
    leaf multi-request{
        type boolean;
        description
            "Support for multiple requests per message.";
```



```
}
leaf gco{
  if-feature gco;
  type boolean;
  description
    "Support for Global Concurrent Optimization
    (GCO).";
}
leaf p2mp{
  if-feature p2mp;
  type boolean;
  description
    "Support for P2MP path computation.";
}

container stateful{
  if-feature stateful;
  description
    "If stateful PCE feature is present";
  leaf enabled{
    type boolean;
    description
      "Enabled or Disabled";
  }
  leaf active{
    type boolean;
    description
      "Support for active stateful PCE.";
  }
  leaf pce-initiated{
    if-feature pce-initiated;
    type boolean;
    description
      "Support for PCE-initiated LSP.";
  }
  leaf include-db-ver{
    if-feature stateful-sync-opt;
    type boolean;
    description
      "Support inclusion of LSP-DB-VERSION
      in LSP object";
  }
  leaf trigger-resync{
    if-feature stateful-sync-opt;
    type boolean;
    description
      "Support PCE triggered re-synchronization";
  }
}
```



```
    leaf trigger-initial-sync{
      if-feature stateful-sync-opt;
      type boolean;
      description
        "PCE triggered initial synchronization";
    }
    leaf incremental-sync{
      if-feature stateful-sync-opt;
      type boolean;
      description
        "Support incremental (delta) sync";
    }
  }
  container sr{
    if-feature sr;
    description
      "If segment routing is supported";
    leaf enabled{
      type boolean;
      description
        "Enabled or Disabled";
    }
  }
}
} //capability

grouping info{
  description
    "This grouping specifies all information which
    maybe relevant to both PCC and PCE.
    This information corresponds to PCE auto-discovery
    information.";
  container domain{
    description
      "The local domain for the PCEP entity";
    list domain{
      key "domain-type domain";
      description
        "The local domain.";
      uses domain{
        description
          "The local domain for the PCEP entity.";
      }
    }
  }
}
container capability{
  description
    "The PCEP entity capability";
```



```
        uses capability{
            description
                "The PCEP entity supported
                capabilities.";
        }
    }

}

} //info

grouping pce-info{
    description
        "This grouping specifies all PCE information
        which maybe relevant to the PCE selection.
        This information corresponds to PCE auto-discovery
        information.";
    container scope{
        description
            "The path computation scope";
        uses pce-scope;
    }

    container neigh-domains{
        description
            "The list of neighbour PCE-Domain
            toward which a PCE can compute
            paths";
        list domain{
            key "domain-type domain";

            description
                "The neighbour domain.";
            uses domain{
                description
                    "The PCE neighbour domain.";
            }
        }
    }
}

} //pce-info

grouping pcep-stats{
    description
        "This grouping defines statistics for PCEP. It is used
        for both peer and current session.";
    leaf avg-rsp-time{
        type uint32;
        units "milliseconds";
        must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
```





```
        " or " +
        "(/pcep-state/entity/peers/peer/role = 'pcc'" +
        " and (. = 0)))" {
    error-message
        "Invalid average response time";
    description
        "If role is pcc then this leaf is meaningless
        and is set to zero.";
}
description
    "The average response time.
    If an average response time has not been
    calculated then this leaf has the value zero.";
}

leaf lwm-rsp-time{
    type uint32;
    units "milliseconds";
    must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
        " or " +
        "(/pcep-state/entity/peers/peer/role = 'pcc'" +
        " and (. = 0)))" {
        error-message
            "Invalid smallest (low-water mark)
            response time";
        description
            "If role is pcc then this leaf is meaningless
            and is set to zero.";
    }
    description
        "The smallest (low-water mark) response time seen.
        If no responses have been received then this
        leaf has the value zero.";
}

leaf hwm-rsp-time{
    type uint32;
    units "milliseconds";
    must "(/pcep-state/entity/peers/peer/role != 'pcc'" +
        " or " +
        "(/pcep-state/entity/peers/peer/role = 'pcc'" +
        " and (. = 0)))" {
        error-message
            "Invalid greatest (high-water mark)
            response time seen";
        description
            "If role is pcc then this field is
            meaningless and is set to zero.";
```



```
    }
    description
      "The greatest (high-water mark) response time seen.
       If no responses have been received then this object
       has the value zero.";
  }

  leaf num-pcreq-sent{
    type yang:counter32;
    description
      "The number of PCReq messages sent.";
  }

  leaf num-pcreq-rcvd{
    type yang:counter32;
    description
      "The number of PCReq messages received.";
  }

  leaf num-pcrep-sent{
    type yang:counter32;
    description
      "The number of PCRep messages sent.";
  }

  leaf num-pcrep-rcvd{
    type yang:counter32;
    description
      "The number of PCRep messages received.";
  }

  leaf num-pcerr-sent{
    type yang:counter32;
    description
      "The number of PCErr messages sent.";
  }

  leaf num-pcerr-rcvd{
    type yang:counter32;
    description
      "The number of PCErr messages received.";
  }

  leaf num-pcntf-sent{
    type yang:counter32;
    description
      "The number of PCNtf messages sent.";
  }
}
```



```
leaf num-pcntf-rcvd{
  type yang:counter32;
  description
    "The number of PCNtf messages received.";
}

leaf num-keepalive-sent{
  type yang:counter32;
  description
    "The number of Keepalive messages sent.";
}

leaf num-keepalive-rcvd{
  type yang:counter32;
  description
    "The number of Keepalive messages received.";
}

leaf num-unknown-rcvd{
  type yang:counter32;
  description
    "The number of unknown messages received.";
}

leaf num-corrupt-rcvd{
  type yang:counter32;
  description
    "The number of corrupted PCEP message received.";
}

leaf num-req-sent{
  type yang:counter32;
  description
    "The number of requests sent. A request corresponds
    1:1 with an RP object in a PCReq message. This might
    be greater than num-pcreq-sent because multiple
    requests can be batched into a single PCReq
    message.";
}

leaf num-req-sent-pend-rep{
  type yang:counter32;
  description
    "The number of requests that have been sent for
    which a response is still pending.";
}

leaf num-req-sent-ero-rcvd{
```



```
    type yang:counter32;
    description
      "The number of requests that have been sent for
       which a response with an ERO object was received.
       Such responses indicate that a path was
       successfully computed by the peer.";
  }

  leaf num-req-sent-nopath-rcvd{
    type yang:counter32;
    description
      "The number of requests that have been sent for
       which a response with a NO-PATH object was
       received. Such responses indicate that the peer
       could not find a path to satisfy the
       request.";
  }

  leaf num-req-sent-cancel-rcvd{
    type yang:counter32;
    description
      "The number of requests that were cancelled with
       a PCNtf message.
       This might be different than num-pcntf-rcvd because
       not all PCNtf messages are used to cancel requests,
       and a single PCNtf message can cancel multiple
       requests.";
  }

  leaf num-req-sent-error-rcvd{
    type yang:counter32;
    description
      "The number of requests that were rejected with a
       PCErr message.
       This might be different than num-pcerr-rcvd because
       not all PCErr messages are used to reject requests,
       and a single PCErr message can reject multiple
       requests.";
  }

  leaf num-req-sent-timeout{
    type yang:counter32;
    description
      "The number of requests that have been sent to a peer
       and have been abandoned because the peer has taken too
       long to respond to them.";
  }
```





```
leaf num-req-sent-cancel-sent{
  type yang:counter32;
  description
    "The number of requests that were sent to the peer and
    explicitly cancelled by the local PCEP entity sending
    a PCNtf.";
}

leaf num-req-rcvd{
  type yang:counter32;
  description
    "The number of requests received. A request
    corresponds 1:1 with an RP object in a PCReq
    message.
    This might be greater than num-pcreq-rcvd because
    multiple requests can be batched into a single
    PCReq message.";
}

leaf num-req-rcvd-pend-rep{
  type yang:counter32;
  description
    "The number of requests that have been received for
    which a response is still pending.";
}

leaf num-req-rcvd-ero-sent{
  type yang:counter32;
  description
    "The number of requests that have been received for
    which a response with an ERO object was sent. Such
    responses indicate that a path was successfully
    computed by the local PCEP entity.";
}

leaf num-req-rcvd-nopath-sent{
  type yang:counter32;
  description
    "The number of requests that have been received for
    which a response with a NO-PATH object was sent. Such
    responses indicate that the local PCEP entity could
    not find a path to satisfy the request.";
}

leaf num-req-rcvd-cancel-sent{
  type yang:counter32;
  description
    "The number of requests received that were cancelled
```



```
        by the local PCEP entity sending a PCNtf message.
        This might be different than num-pcntf-sent because
        not all PCNtf messages are used to cancel requests,
        and a single PCNtf message can cancel multiple
        requests.";
    }

    leaf num-req-rcvd-error-sent{
        type yang:counter32;
        description
            "The number of requests received that were cancelled
            by the local PCEP entity sending a PCErr message.
            This might be different than num-pcerr-sent because
            not all PCErr messages are used to cancel requests,
            and a single PCErr message can cancel multiple
            requests.";
    }

    leaf num-req-rcvd-cancel-rcvd{
        type yang:counter32;
        description
            "The number of requests that were received from the
            peer and explicitly cancelled by the peer sending
            a PCNtf.";
    }

    leaf num-rep-rcvd-unknown{
        type yang:counter32;
        description
            "The number of responses to unknown requests
            received. A response to an unknown request is a
            response whose RP object does not contain the
            request ID of any request that is currently
            outstanding on the session.";
    }

    leaf num-req-rcvd-unknown{
        type yang:counter32;
        description
            "The number of unknown requests that have been
            received. An unknown request is a request
            whose RP object contains a request ID of
            zero.";
    }

    container svec{
        if-feature svec;
        description
```



```
        "If synchronized path computation is supported";
    leaf num-svec-sent{
        type yang:counter32;
        description
            "The number of SVEC objects sent in PCReq messages.
            An SVEC object represents a set of synchronized
            requests.";
    }

    leaf num-svec-req-sent{
        type yang:counter32;
        description
            "The number of requests sent that appeared in one
            or more SVEC objects.";
    }

    leaf num-svec-rcvd{
        type yang:counter32;
        description
            "The number of SVEC objects received in PCReq
            messages. An SVEC object represents a set of
            synchronized requests.";
    }

    leaf num-svec-req-rcvd{
        type yang:counter32;
        description
            "The number of requests received that appeared
            in one or more SVEC objects.";
    }
}
container stateful{
    if-feature stateful;
    description
        "Stateful PCE related statistics";
    leaf num-pcrpt-sent{
        type yang:counter32;
        description
            "The number of PCRpt messages sent.";
    }

    leaf num-pcrpt-rcvd{
        type yang:counter32;
        description
            "The number of PCRpt messages received.";
    }

    leaf num-pcupd-sent{
```



```
    type yang:counter32;
    description
      "The number of PCUpd messages sent.";
  }

  leaf num-pcupd-rcvd{
    type yang:counter32;
    description
      "The number of PCUpd messages received.";
  }

  leaf num-rpt-sent{
    type yang:counter32;
    description
      "The number of LSP Reports sent.  A LSP report
       corresponds 1:1 with an LSP object in a PCRpt
       message. This might be greater than
       num-pcrpt-sent because multiple reports can
       be batched into a single PCRpt message.";
  }

  leaf num-rpt-rcvd{
    type yang:counter32;
    description
      "The number of LSP Reports received.  A LSP report
       corresponds 1:1 with an LSP object in a PCRpt
       message.
       This might be greater than num-pcrpt-rcvd because
       multiple reports can be batched into a single
       PCRpt message.";
  }

  leaf num-rpt-rcvd-error-sent{
    type yang:counter32;
    description
      "The number of reports of LSPs received that were
       responded by the local PCEP entity by sending a
       PCErr message.";
  }

  leaf num-upd-sent{
    type yang:counter32;
    description
      "The number of LSP updates sent.  A LSP update
       corresponds 1:1 with an LSP object in a PCUpd
       message. This might be greater than
       num-pcupd-sent because multiple updates can
       be batched into a single PCUpd message.";
```





```
}

leaf num-upd-rcvd{
  type yang:counter32;
  description
    "The number of LSP Updates received. A LSP update
    corresponds 1:1 with an LSP object in a PCUpd
    message.
    This might be greater than num-pcupd-rcvd because
    multiple updates can be batched into a single
    PCUpd message.";
}

leaf num-upd-rcvd-unknown{
  type yang:counter32;
  description
    "The number of updates to unknown LSPs
    received. An update to an unknown LSP is a
    update whose LSP object does not contain the
    PLSP-ID of any LSP that is currently
    present.";
}

leaf num-upd-rcvd-undelegated{
  type yang:counter32;
  description
    "The number of updates to not delegated LSPs
    received. An update to an undelegated LSP is a
    update whose LSP object does not contain the
    PLSP-ID of any LSP that is currently
    delegated to current PCEP session.";
}

leaf num-upd-rcvd-error-sent{
  type yang:counter32;
  description
    "The number of updates to LSPs received that were
    responded by the local PCEP entity by sending a
    PCErr message.";
}

container initiation {
  if-feature pce-initiated;
  description
    "PCE-Initiated related statistics";
  leaf num-pcinitiate-sent{
    type yang:counter32;
    description
      "The number of PCInitiate messages sent.";
```



```
    }

    leaf num-pcinitiate-rcvd{
      type yang:counter32;
      description
        "The number of PCInitiate messages received.";
    }

    leaf num-initiate-sent{
      type yang:counter32;
      description
        "The number of LSP Initiation sent via PCE.
        A LSP initiation corresponds 1:1 with an LSP
        object in a PCInitiate message. This might be
        greater than num-pcinitiate-sent because
        multiple initiations can be batched into a
        single PCInitiate message.";
    }

    leaf num-initiate-rcvd{
      type yang:counter32;
      description
        "The number of LSP Initiation received from
        PCE. A LSP initiation corresponds 1:1 with
        an LSP object in a PCInitiate message. This
        might be greater than num-pcinitiate-rcvd
        because multiple initiations can be batched
        into a single PCInitiate message.";
    }

    leaf num-initiate-rcvd-error-sent{
      type yang:counter32;
      description
        "The number of initiations of LSPs received
        that were responded by the local PCEP entity
        by sending a PCErr message.";
    }
  }
}

container path-key {
  if-feature path-key;
  description
    "If Path-Key is supported";
  leaf num-unknown-path-key{
    type yang:counter32;
    description
      "The number of attempts to expand an unknown
      path-key.";
```



```
    }
    leaf num-exp-path-key{
      type yang:counter32;
      description
        "The number of attempts to expand an expired
        path-key.";
    }
    leaf num-dup-path-key{
      type yang:counter32;
      description
        "The number of duplicate attempts to expand same
        path-key.";
    }
    leaf num-path-key-no-attempt{
      type yang:counter32;
      description
        "The number of expired path-keys with no attempt to
        expand it.";
    }
  }
}
} //pcep-stats

grouping lsp-state{
  description
    "This grouping defines the attributes for LSP in LSP-DB.
    These are the attributes specifically from the PCEP
    perspective";
  leaf plsp-id{
    type uint32{
      range "1..1048575";
    }
    description
      "A PCEP-specific identifier for the LSP. A PCC
      creates a unique PLSP-ID for each LSP that is
      constant for the lifetime of a PCEP session.
      PLSP-ID is 20 bits with 0 and 0xFFFFF are
      reserved";
  }
  leaf pcc-id{
    type inet:ip-address;
    description
      "The local internet address of the PCC, that
      generated the PLSP-ID.";
  }
}

container lsp-ref{
  description
    "reference to ietf-te lsp state";
}
```



```
leaf source {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:source";
  }
  description
    "Tunnel sender address extracted from
    SENDER_TEMPLATE object";
  reference "RFC3209";
}
leaf destination {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:"
      + "destination";
  }
  description
    "Tunnel endpoint address extracted from
    SESSION object";
  reference "RFC3209";
}
leaf tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:tunnel-id";
  }
  description
    "Tunnel identifier used in the SESSION
    that remains constant over the life
    of the tunnel.";
  reference "RFC3209";
}
leaf lsp-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:lsp-id";
  }
  description
    "Identifier used in the SENDER_TEMPLATE
    and the FILTER_SPEC that can be changed
    to allow a sender to share resources with
    itself.";
  reference "RFC3209";
}
leaf extended-tunnel-id {
  type leafref {
    path "/te:te/te:lsps-state/te:lsp/te:"
      + "extended-tunnel-id";
  }
  description
    "Extended Tunnel ID of the LSP.";
  reference "RFC3209";
}
```





```
    }
    leaf type {
      type leafref {
        path "/te:te/te:lsps-state/te:lsp/te:type";
      }
      description "LSP type P2P or P2MP";
    }
  }

  leaf admin-state{
    type boolean;
    description
      "The desired operational state";
  }
  leaf operational-state{
    type operational-state;
    description
      "The operational status of the LSP";
  }
  container delegated{
    description
      "The delegation related parameters";
    leaf enabled{
      type boolean;
      description
        "LSP is delegated or not";
    }
    leaf pce{
      type leafref {
        path "/pcep-state/entity/peers/peer/addr";
      }
      must "(../enabled = true())"
      {
        error-message
          "The LSP must be delegated";
        description
          "When LSP is a delegated LSP";
      }
      description
        "The reference to the PCE peer to
        which LSP is delegated";
    }
  }
  leaf srp-id{
    type uint32;
    description
      "The last SRP-ID-number associated with this
      LSP.";
  }
}
```



```
    }
    container initiation {
      if-feature pce-initiated;
      description
        "The PCE initiation related parameters";
      leaf enabled{
        type boolean;
        description
          "LSP is PCE-initiated or not";
      }
      leaf pce{
        type leafref {
          path "/pcep-state/entity/peers/peer/addr";
        }
        must "(../enabled = true())"
        {
          error-message
            "The LSP must be PCE-Initiated";
          description
            "When the LSP must be PCE-Initiated";
        }
        description
          "The reference to the PCE
            that initiated this LSP";
      }
    }
  }
  leaf symbolic-path-name{
    type string;
    description
      "The symbolic path name associated with the LSP.";
  }
  leaf last-error{
    type lsp-error;
    description
      "The last error for the LSP.";
  }
  leaf pst{
    type pst;
    default "rsvp-te";
    description
      "The Path Setup Type";
  }
}

} //lsp-state

grouping notification-instance-hdr {
  description
```



```
        "This group describes common instance specific data
        for notifications.";

    leaf peer-addr {
        type leafref {
            path "/pcep-state/entity/peers/peer/addr";
        }
        description
            "Reference to peer address";
    }

}

} // notification-instance-hdr

grouping notification-session-hdr {
    description
        "This group describes common session instance specific
        data for notifications.";

    leaf session-initiator {
        type leafref {
            path "/pcep-state/entity/peers/peer/sessions/" +
                "session/initiator";
        }
        description
            "Reference to pcep session initiator leaf";
    }
}

} // notification-session-hdr

grouping stateful-pce-parameter {
    description
        "This group describes stateful PCE specific
        parameters.";
    leaf state-timeout {
        type uint32;
        units "seconds";
        description
            "When a PCEP session is terminated, a PCC
            waits for this time period before flushing
            LSP state associated with that PCEP session
            and reverting to operator-defined default
            parameters or behaviours.";
    }
    leaf redelegation-timeout {
        type uint32;
        units "seconds";
        must "(/pcep-state/entity/role = 'pcc')" +
            " or " +
            "(/pcep-state/entity/role = 'pcc-and-pce'))" +
```



```
        " and " +
        "(/pcep/entity/capability/stateful/active"
        + "= true())"
    {
        error-message "The PCEP entity must be PCC";
        description
            "When PCEP entity is PCC";
    }
    description
        "When a PCEP session is terminated, a PCC
        waits for this time period before revoking
        LSP delegation to a PCE and attempting to
        redelegate LSPs associated with the
        terminated PCEP session to an alternate
        PCE.";
}
leaf rpt-non-pcep-lsp{
    type boolean;
    must "(/pcep-state/entity/role = 'pcc')" +
        " or " +
        "(/pcep-state/entity/role = 'pcc-and-pce'))"
    {
        error-message "The PCEP entity must be PCC";
        description
            "When PCEP entity is PCC";
    }
    description
        "If set, a PCC reports LSPs that are not
        controlled by any PCE (for example, LSPs
        that are statically configured at the
        PCC). ";
}
}

grouping authentication {
    description "Authentication Information";
    choice auth-type-selection {
        description
            "Options for expressing authentication setting.";
        case auth-key-chain {
            leaf key-chain {
                type key-chain:key-chain-ref;
                description
                    "key-chain name.";
            }
        }
        case auth-key {
```





```
        leaf key {
            type string;
            description
                "Key string in ASCII format.";
        }
        container crypto-algorithm {
            uses key-chain:crypto-algorithm-types;
            description
                "Cryptographic algorithm associated
                 with key.";
        }
    }
    case auth-tls {
        if-feature tls;
        container tls {
            description
                "TLS related information - TBD";
        }
    }
}

grouping path-key {
    description "Path-key related information";
    leaf enabled{
        type boolean;
        description
            "Enabled or Disabled";
    }
    leaf discard-timer {
        type uint32;
        units "minutes";
        default 10;
        description
            "A timer to discard unwanted path-keys";
    }
    leaf reuse-time {
        type uint32;
        units "minutes";
        default 30;
        description
            "A time after which the path-keys could be reused";
    }
    leaf pce-id {
        type inet:ip-address;
        description
            "PCE Address to be used in each Path-Key Subobject
             (PKS)";
    }
}
```



```
    }
  }

  grouping path-key-state {
    description "Table to allow inspection of path-keys";
    list path-keys{
      key "path-key";

      description
        "The list of path-keys generated by the PCE";

      leaf path-key {
        type uint16;
        description
          "The identifier, or token used to represent
           the Confidential Path Segment (CPS) within
           the context of the PCE";
      }
    }
    container cps {
      description
        "The Confidential Path Segment (CPS)";
      list explicit-route-objects {
        key "index";
        description
          "List of explicit route objects";
        leaf index {
          type uint8 {
            range "0..255";
          }
          description
            "Index of this explicit route object";
        }
        leaf explicit-route-usage {
          type identityref {
            base te-types:route-usage-type;
          }
          description
            "An explicit-route hop action.";
        }
        uses te-types:explicit-route-subobject;
      }
    }
    leaf pcc-original {
      type leafref {
        path "/pcep-state/entity/peers/peer/addr";
      }
      description
        "Reference to PCC peer address of
```



```
        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-state/entity/peers/peer/addr";
        }
        must "(../retrieved = true())"
        {
            error-message
                "The Path-key should be retrieved";
            description
                "When Path-Key has been retrieved";
        }
        description
            "Reference to PCC peer address which
            retrieved the path-key";
    }
    leaf creation-time {
        type yang:timestamp;
        description
            "The timestamp value at the time this Path-Key was
            created.";
    }
    leaf discard-time {
        type uint32;
        units "minutes";
        description
            "A time after which this path-keys will be
            discarded";
    }
    leaf reuse-time {
        type uint32;
        units "minutes";
        description
            "A time after which this path-keys could be
            reused";
    }
}
```



```
}

grouping of-list {
  description "List of OF";
  list objective-function{
    key "of";

    description
      "The list of authorized OF";

    leaf of {
      type objective-function;
      description
        "The OF authorized";
    }
  }
}

grouping association {
  description
    "Generic Association parameters";
  leaf type {
    type "assoc-type";
    description
      "The PCEP association type";
  }
  leaf id {
    type uint16;
    description
      "PCEP Association ID";
  }
  leaf source {
    type inet:ip-address;
    description
      "PCEP Association Source.";
  }
  leaf global-source {
    type uint32;
    description
      "PCEP Association Global
      Source.";
  }
  leaf extended-id{
    type string;
    description
      "Additional information to
      support unique identification.";
  }
}
```





```
grouping association-ref {
  description
    "Generic Association parameters";
  leaf id {
    type leafref {
      path "/pcep-state/entity/lsp-db/"
        + "association-list/id";
    }
    description
      "PCEP Association ID";
  }
  leaf source {
    type leafref {
      path "/pcep-state/entity/lsp-db/"
        + "association-list/source";
    }
    description
      "PCEP Association Source.";
  }
  leaf global-source {
    type leafref {
      path "/pcep-state/entity/lsp-db/"
        + "association-list/global-source";
    }
    description
      "PCEP Association Global
      Source.";
  }
  leaf extended-id{
    type leafref {
      path "/pcep-state/entity/lsp-db/"
        + "association-list/extended-id";
    }
    description
      "Additional information to
      support unique identification.";
  }
}
/*
 * Configuration data nodes
 */
container pcep{

  presence
    "The PCEP is enabled";

  description
    "Parameters for list of configured PCEP entities"
```



```
        on the device.";

container entity {

    description
        "The configured PCEP entity on the device.";

    leaf addr {
        type inet:ip-address;
        mandatory true;
        description
            "The local Internet address of this PCEP
            entity.
            If operating as a PCE server, the PCEP
            entity listens on this address.
            If operating as a PCC, the PCEP entity
            binds outgoing TCP connections to this
            address.
            It is possible for the PCEP entity to
            operate both as a PCC and a PCE Server, in
            which case it uses this address both to
            listen for incoming TCP connections and to
            bind outgoing TCP connections.";
    }

    leaf enabled {
        type boolean;
        default true;
        description
            "The administrative status of this PCEP
            Entity.";
    }

    leaf role {
        type pcep-role;
        mandatory true;
        description
            "The role that this entity can play.
            Takes one of the following values.
            - unknown(0): this PCEP Entity role is not
              known.
            - pcc(1): this PCEP Entity is a PCC.
            - pce(2): this PCEP Entity is a PCE.
            - pcc-and-pce(3): this PCEP Entity is both
              a PCC and a PCE.";
    }

    leaf description {
```



```
    type string;
    description
        "Description of the PCEP entity configured
        by the user";
}

leaf speaker-entity-id{
    if-feature stateful-sync-opt;
    type string;
    description
        "The Speaker Entity Identifier";
}

uses info {
    description
        "Local PCEP entity information";
}

container pce-info {
    must "(/pcep-state/entity/role = 'pce')" +
        " or " +
        "(/pcep-state/entity/role = 'pcc-and-pce'))"
    {
        error-message "The PCEP entity must be PCE";
        description
            "When PCEP entity is PCE";
    }
    uses pce-info {
        description
            "Local PCE information";
    }
    container path-key {
        if-feature path-key;
        uses path-key {
            description
                "Path-Key Configuration";
        }
        description
            "Path-Key Configuration";
    }

    description
        "The Local PCE Entity PCE information";
}

uses authentication {
    description
```



```
        "Local PCEP entity authentication information";
    }

    uses pcep-entity-info {
        description
            "The configuration related to the PCEP
            entity.";
    }

    leaf pcep-notification-max-rate {
        type uint32;
        mandatory true;
        description
            "This variable indicates the maximum number of
            notifications issued per second. If events occur
            more rapidly, the implementation may simply fail
            to emit these notifications during that period,
            or may queue them until an appropriate time. A
            value of 0 means no notifications are emitted
            and all should be discarded (that is, not
            queued).";
    }

    container stateful-parameter{
        if-feature stateful;
        must "(/pcep/entity/capability/stateful/enabled" +
            " = true())"
        {
            error-message
                "The Stateful PCE must be enabled";
            description
                "When PCEP entity is stateful
                enabled";
        }
        uses stateful-pce-parameter;

        description
            "The configured stateful parameters";
    }

    container of-list{
        if-feature objective-function;
        must "(/pcep/entity/role = 'pce')" +
            " or " +
            "(/pcep/entity/role = 'pcc-and-pce'))"
```





```
{
  error-message
    "The PCEP entity must be PCE";
  description
    "The authorized OF-List at PCE";
}
uses of-list;

description
  "The authorized OF-List at PCE for all peers";
}

container peers{
  must "(/pcep/entity/role = 'pcc')" +
    " or " +
    "(/pcep/entity/role = 'pcc-and-pce'))"
  {
    error-message
      "The PCEP entity must be PCC";
    description
      "When PCEP entity is PCC, as remote
        PCE peers are configured.";
  }
  description
    "The list of configured peers for the
      entity (remote PCE)";
  list peer{
    key "addr";

    description
      "The peer configured for the entity.
        (remote PCE)";

    leaf addr {
      type inet:ip-address;
      description
        "The local Internet address of this
          PCEP peer.";
    }

    leaf description {
      type string;
      description
        "Description of the PCEP peer
          configured by the user";
    }
  }
  uses info {
```



```
        description
            "PCE Peer information";
    }
    uses pce-info {
        description
            "PCE Peer information";
    }

    leaf delegation-pref{
        if-feature stateful;
        type uint8{
            range "0..7";
        }
        must "(/pcep/entity/capability/stateful/active"
            + "= true())"
        {
            error-message
                "The Active Stateful PCE must be
                enabled";
            description
                "When PCEP entity is active stateful
                enabled";
        }
        description
            "The PCE peer delegation preference.";
    }
    uses authentication {
        description
            "PCE Peer authentication";
    }
    container of-list{
        if-feature objective-function;
        must "(/pcep/entity/role = 'pce')" +
            " or " +
            "(/pcep/entity/role = 'pcc-and-pce'))"
        {
            error-message
                "The PCEP entity must be PCE";
            description
                "The authorized OF-List at PCE";
        }
        uses of-list;

        description
            "The authorized OF-List a specific peer";
    }
} //peer
} //peers
```



```
    }//entity
  }//pcep

  /*
   * Operational data nodes
   */

  container pcep-state{
    config false;
    description
      "The list of operational PCEP entities on the
      device.";

    container entity{
      description
        "The operational PCEP entity on the device.";

      leaf addr {
        type inet:ip-address;
        description
          "The local Internet address of this PCEP
          entity.
          If operating as a PCE server, the PCEP
          entity listens on this address.
          If operating as a PCC, the PCEP entity
          binds outgoing TCP connections to this
          address.
          It is possible for the PCEP entity to
          operate both as a PCC and a PCE Server, in
          which case it uses this address both to
          listen for incoming TCP connections and to
          bind outgoing TCP connections.";
      }

      leaf index{
        type uint32;
        description
          "The index of the operational PCEP
          entity";
      }

      leaf admin-status {
        type pcep-admin-status;
        description
          "The administrative status of this PCEP Entity.
          This is the desired operational status as
          currently set by an operator or by default in
```



```
        the implementation. The value of enabled
        represents the current status of an attempt
        to reach this desired status.";
    }

    leaf oper-status {
        type pcep-admin-status;
        description
            "The operational status of the PCEP entity.
            Takes one of the following values.
            - oper-status-up(1): the PCEP entity is
              active.
            - oper-status-down(2): the PCEP entity is
              inactive.
            - oper-status-going-up(3): the PCEP entity is
              activating.
            - oper-status-going-down(4): the PCEP entity is
              deactivating.
            - oper-status-failed(5): the PCEP entity has
              failed and will recover when possible.
            - oper-status-failed-perm(6): the PCEP entity
              has failed and will not recover without
              operator intervention.";
    }

    leaf role {
        type pcep-role;
        description
            "The role that this entity can play.
            Takes one of the following values.
            - unknown(0): this PCEP entity role is
              not known.
            - pcc(1): this PCEP entity is a PCC.
            - pce(2): this PCEP entity is a PCE.
            - pcc-and-pce(3): this PCEP entity is
              both a PCC and a PCE.";
    }

    leaf description {
        type string;
        description
            "Description of the PCEP entity configured
            by the user";
    }

    leaf speaker-entity-id{
        if-feature stateful-sync-opt;
        type string;
```





```
        description
            "The Speaker Entity Identifier";
    }

    uses info {
        description
            "Local PCEP entity information";
    }

    container pce-info {
        when "(/pcep-state/entity/role = 'pce')" +
            " or " +
            "(/pcep-state/entity/role = 'pcc-and-pce'))"
        {
            description
                "When PCEP entity is PCE";
        }
        uses pce-info {
            description
                "Local PCE information";
        }

        container path-key {
            if-feature path-key;
            uses path-key {
                description
                    "Path-Key Configuration";
            }
            description
                "Path-Key Configuration";
        }

        description
            "The Local PCE Entity PCE information";
    }

    uses authentication {
        description
            "Local PCEP Entity authentication information";
    }

    uses pcep-entity-info{
        description
            "The operational information related to the
            PCEP entity.";
    }

    container stateful-parameter{
```



```
    if-feature stateful;
    must "(/pcep/entity/capability/stateful/enabled" +
        " = true())"
    {
        error-message
            "The Stateful PCE must be enabled";
        description
            "When PCEP entity is stateful
            enabled";
    }
    uses stateful-pce-parameter;

    description
        "The operational stateful parameters";
}
```

```
container lsp-db{
    if-feature stateful;
    description
        "The LSP-DB";
    leaf db-ver{
        if-feature stateful-sync-opt;
        type uint64;
        must "((/pcep/entity/role = 'pcc')" +
            " or " +
            "(/pcep/entity/role = 'pcc-and-pce'))"
        {
            error-message
                "The PCEP entity must be PCC";
            description
                "When PCEP entity is PCC, as remote
                PCE peers are configured.";
        }

        description
            "The LSP State Database Version Number";
    }
    list association-list {
        key "id source global-source extended-id";
        description
            "List of all PCEP associations";
        uses association {
            description
                "The Association attributes";
        }
        list lsp {
```



```
        key "plsp-id pcc-id";
        description
            "List of all LSP in this association";
        leaf plsp-id {
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                    + "lsp/plsp-id";
            }
            description
                "Reference to PLSP-ID in LSP-DB";
        }
        leaf pcc-id {
            type leafref {
                path "/pcep-state/entity/lsp-db/"
                    + "lsp/pcc-id";
            }
            description
                "Reference to PCC-ID in LSP-DB";
        }
    }
}
list lsp{
    key "plsp-id pcc-id";
    description
        "List of all LSPs in LSP-DB";
    uses lsp-state{
        description
            "The PCEP specific attributes for
            LSP-DB.";
    }
    list association-list {
        key "id source global-source extended-id";
        description
            "List of all PCEP associations";
        uses association-ref {
            description
                "Reference to the Association
                attributes";
        }
    }
}
}
container path-keys {
    if-feature path-key;
    must "((/pcep-state/entity/role = 'pce'))" +
        " or " +
        "(/pcep-state/entity/role = 'pcc-and-pce'))"
```



```
{
  error-message
    "The PCEP entity must be PCE";
  description
    "When PCEP entity is PCE";
}
uses path-key-state;
description
  "The path-keys generated by the PCE";
}
container of-list{
  if-feature objective-function;
  must "(/pcep/entity/role = 'pce')\" +
    \" or \" +
    \"(/pcep/entity/role = 'pcc-and-pce'))\"
  {
    error-message
      "The PCEP entity must be PCE";
    description
      "The authorized OF-List at PCE";
  }
  uses of-list;

  description
    "The authorized OF-List at PCE for all peers";
}
container peers{
  description
    "The list of peers for the entity";

  list peer{
    key "addr";

    description
      "The peer for the entity.";

    leaf addr {
      type inet:ip-address;
      description
        "The local Internet address of this PCEP
        peer.";
    }

    leaf role {
      type pcep-role;
      description
        "The role of the PCEP Peer.
        Takes one of the following values.
```





```
        - unknown(0): this PCEP peer role
          is not known.
        - pcc(1): this PCEP peer is a PCC.
        - pce(2): this PCEP peer is a PCE.
        - pcc-and-pce(3): this PCEP peer
          is both a PCC and a PCE.";
    }

    uses info {
        description
            "PCEP peer information";
    }

    container pce-info {
        when "(/pcep-state/entity/role = 'pcc')" +
            " or " +
            "(/pcep-state/entity/role " +
            "= 'pcc-and-pce'))"
        {
            description
                "When PCEP entity is PCE";
        }
        uses pce-info {
            description
                "PCE Peer information";
        }
        description
            "The PCE Peer information";
    }

    leaf delegation-pref{
        if-feature stateful;
        type uint8{
            range "0..7";
        }
        must "(/pcep-state/entity/role = 'pcc')" +
            " or " +
            "(/pcep-state/entity/role" +
            " = 'pcc-and-pce'))"
        {
            error-message
                "The PCEP entity must be PCC";
            description
                "When PCEP entity is PCC";
        }
        must "(/pcep/entity/capability/stateful/active"
```



```
        + "= true()")
    {
        error-message
            "The Active Stateful PCE must be
            enabled";
        description
            "When PCEP entity is active stateful
            enabled";
    }
    description
        "The PCE peer delegation preference.";
}

uses authentication {
    description
        "PCE Peer authentication";
}

container of-list{
    if-feature objective-function;
    must "(/pcep/entity/role = 'pce') " +
    " or " +
    "(/pcep/entity/role = 'pcc-and-pce'))"
    {
        error-message
            "The PCEP entity must be PCE";
        description
            "The authorized OF-List at PCE";
    }
    uses of-list;

    description
        "The authorized OF-List of a specific
        peer";
}

leaf discontinuity-time {
    type yang:timestamp;
    description
        "The timestamp of the time when the
        information and statistics were
        last reset.";
}

leaf initiate-session {
    type boolean;
    description
        "Indicates whether the local PCEP
        entity initiates sessions to this peer,
```



```
        or waits for the peer to initiate a
        session.";
    }

    leaf session-exists{
        type boolean;
        description
            "Indicates whether a session with
            this peer currently exists.";
    }

    leaf num-sess-setup-ok{
        type yang:counter32;
        description
            "The number of PCEP sessions successfully
            successfully established with the peer,
            including any current session. This
            counter is incremented each time a
            session with this peer is successfully
            established.";
    }

    leaf num-sess-setup-fail{
        type yang:counter32;
        description
            "The number of PCEP sessions with the peer
            that have been attempted but failed
            before being fully established. This
            counter is incremented each time a
            session retry to this peer fails.";
    }

    leaf session-up-time{
        type yang:timestamp;
        must "(../num-sess-setup-ok != 0 or " +
            "(../num-sess-setup-ok = 0 and " +
            "(. = 0)))" {
            error-message
                "Invalid Session Up timestamp";
            description
                "If num-sess-setup-ok is zero,
                then this leaf contains zero.";
        }
        description
            "The timestamp value of the last time a
            session with this peer was successfully
            established.";
    }
}
```



```
leaf session-fail-time{
  type yang:timestamp;
  must "(../num-sess-setup-fail != 0 or " +
    "(../num-sess-setup-fail = 0 and " +
    "(. = 0)))" {
    error-message
      "Invalid Session Fail timestamp";
    description
      "If num-sess-setup-fail is zero,
        then this leaf contains zero.";
  }
  description
    "The timestamp value of the last time a
      session with this peer failed to be
      established.";
}

leaf session-fail-up-time{
  type yang:timestamp;
  must "(../num-sess-setup-ok != 0 or " +
    "(../num-sess-setup-ok = 0 and " +
    "(. = 0)))" {
    error-message
      "Invalid Session Fail from
        Up timestamp";
    description
      "If num-sess-setup-ok is zero,
        then this leaf contains zero.";
  }
  description
    "The timestamp value of the last time a
      session with this peer failed from
      active.";
}

container pcep-stats {
  description
    "The container for all statistics at peer
      level.";
  uses pcep-stats{
    description
      "Since PCEP sessions can be
        ephemeral, the peer statistics tracks
        a peer even when no PCEP session
        currently exists to that peer. The
        statistics contained are an aggregate
        of the statistics for all successive
        sessions to that peer.";
```





```
    }

    leaf num-req-sent-closed{
      type yang:counter32;
      description
        "The number of requests that were
         sent to the peer and implicitly
         cancelled when the session they were
         sent over was closed.";
    }

    leaf num-req-rcvd-closed{
      type yang:counter32;
      description
        "The number of requests that were
         received from the peer and
         implicitly cancelled when the
         session they were received over
         was closed.";
    }
  }//pcep-stats
```

```
container sessions {
  description
    "This entry represents a single PCEP
     session in which the local PCEP entity
     participates.
     This entry exists only if the
     corresponding PCEP session has been
     initialized by some event, such as
     manual user configuration, auto-
     discovery of a peer, or an incoming
     TCP connection.";

  list session {
    key "initiator";

    description
      "The list of sessions, note that
       for a time being two sessions
       may exist for a peer";

    leaf initiator {
      type pcep-initiator;
      description
        "The initiator of the session,
```



that is, whether the TCP connection was initiated by the local PCEP entity or the peer.

There is a window during session initialization where two sessions can exist between a pair of PCEP speakers, each initiated by one of the speakers. One of these sessions is always discarded before it leaves OpenWait state. However, before it is discarded, two sessions to the given peer appear transiently in this MIB module. The sessions are distinguished by who initiated them, and so this field is the key.";

}

```
leaf state-last-change {  
  type yang:timestamp;  
  description  
    "The timestamp value at the  
    time this session entered its  
    current state as denoted by  
    the state leaf.";
```

```
}
```

```
leaf state {  
  type pcep-sess-state;  
  description  
    "The current state of the  
    session.  
    The set of possible states  
    excludes the idle state since  
    entries do not exist in the  
    idle state.";
```

```
}
```

```
leaf session-creation {  
  type yang:timestamp;  
  description  
    "The timestamp value at the  
    time this session was  
    created.";
```

```
}
```



```
leaf connect-retry {
  type yang:counter32;
  description
    "The number of times that the
     local PCEP entity has
     attempted to establish a TCP
     connection for this session
     without success. The PCEP
     entity gives up when this
     reaches connect-max-retry.";
}

leaf local-id {
  type uint32 {
    range "0..255";
  }
  description
    "The value of the PCEP session
     ID used by the local PCEP
     entity in the Open message
     for this session.
     If state is tcp-pending then
     this is the session ID that
     will be used in the Open
     message. Otherwise, this is
     the session ID that was sent
     in the Open message.";
}

leaf remote-id {
  type uint32 {
    range "0..255";
  }
  must "((../state != 'tcp-pending'" +
    "and " +
    "../state != 'open-wait' )" +
    "or " +
    "((../state = 'tcp-pending'" +
    " or " +
    "../state = 'open-wait' )" +
    "and (. = 0)))" {
    error-message
      "Invalid remote-id";
    description
      "If state is tcp-pending
       or open-wait then this
       leaf is not used and
       MUST be set to zero.";
```



```
    }
    description
      "The value of the PCEP session
       ID used by the peer in its
       Open message for this
       session.";
  }

  leaf keepalive-timer {
    type uint32 {
      range "0..255";
    }
    units "seconds";
    must "(../state = 'session-up'" +
      "or " +
      "(../state != 'session-up'" +
      "and (. = 0)))" {
      error-message
        "Invalid keepalive
         timer";
      description
        "This field is used if
         and only if state is
         session-up. Otherwise,
         it is not used and
         MUST be set to
         zero.";
    }
    description
      "The agreed maximum interval at
       which the local PCEP entity
       transmits PCEP messages on this
       PCEP session. Zero means that
       the local PCEP entity never
       sends Keepalives on this
       session.";
  }

  leaf peer-keepalive-timer {
    type uint32 {
      range "0..255";
    }
    units "seconds";
    must "(../state = 'session-up'" +
      "or " +
      "(../state != 'session-up'" +
      "and " +
      "(. = 0)))" {
```





```
        error-message
            "Invalid Peer keepalive
            timer";
        description
            "This field is used if
            and only if state is
            session-up. Otherwise,
            it is not used and MUST
            be set to zero.";
    }
    description
        "The agreed maximum interval at
        which the peer transmits PCEP
        messages on this PCEP session.
        Zero means that the peer never
        sends Keepalives on this
        session.";
}

leaf dead-timer {
    type uint32 {
        range "0..255";
    }
    units "seconds";
    description
        "The dead timer interval for
        this PCEP session.";
}

leaf peer-dead-timer {
    type uint32 {
        range "0..255";
    }
    units "seconds";
    must "((../state != 'tcp-pending' " +
        "and " +
        "../state != 'open-wait' )" +
        "or " +
        "((../state = 'tcp-pending' " +
        " or " +
        "../state = 'open-wait' )" +
        "and " +
        "(. = 0)))" {
        error-message
            "Invalid Peer Dead
            timer";
        description
            "If state is tcp-
```



```
        pending or open-wait
        then this leaf is not
        used and MUST be set to
        zero.";
    }
    description
        "The peer's dead-timer interval
        for this PCEP session.";
}

leaf ka-hold-time-rem {
    type uint32 {
        range "0..255";
    }
    units "seconds";
    must "((../state != 'tcp-pending'" +
        "and " +
        "../state != 'open-wait' ) " +
        "or " +
        "((../state = 'tcp-pending'" +
        "or " +
        "../state = 'open-wait' )" +
        "and " +
        "(. = 0)))" {
        error-message
            "Invalid Keepalive hold
            time remaining";
        description
            "If state is tcp-pending
            or open-wait then this
            field is not used and
            MUST be set to zero.";
    }
    description
        "The keep alive hold time
        remaining for this session.";
}

leaf overloaded {
    type boolean;
    description
        "If the local PCEP entity has
        informed the peer that it is
        currently overloaded, then this
        is set to true. Otherwise, it
        is set to false.";
}
```



```
leaf overload-time {
  type uint32;
  units "seconds";
  must "((../overloaded = true()) or" +
    "((../overloaded != true()) and" +
    " (. = 0)))" {
    error-message
      "Invalid overload-time";
    description
      "This field is only used
      if overloaded is set to
      true. Otherwise, it is
      not used and MUST be set
      to zero.";
  }
  description
    "The interval of time that is
    remaining until the local PCEP
    entity will cease to be
    overloaded on this session.";
}

leaf peer-overloaded {
  type boolean;
  description
    "If the peer has informed the
    local PCEP entity that it is
    currently overloaded, then this
    is set to true. Otherwise, it
    is set to false.";
}

leaf peer-overload-time {
  type uint32;
  units "seconds";
  must "((../peer-overloaded = true())" +
    " or " +
    "((../peer-overloaded != true())" +
    " and " +
    " (. = 0)))" {
    error-message
      "Invalid peer overload
      time";
    description
      "This field is only used
      if peer-overloaded is
      set to true. Otherwise,
      it is not used and MUST
```



```
        be set to zero.";
    }
    description
        "The interval of time that is
        remaining until the peer will
        cease to be overloaded. If it
        is not known how long the peer
        will stay in overloaded state,
        this leaf is set to zero.";
}
leaf lspdb-sync {
    if-feature stateful;
    type sync-state;
    description
        "The LSP-DB state synchronization
        status.";
}
leaf recv-db-ver{
    if-feature stateful;
    if-feature stateful-sync-opt;
    type uint64;
    must "(/pcep-state/entity/peers/" +
        "peer/role = 'pcc')" +
        " or " +
        "(/pcep-state/entity/peers/" +
        "peer/role = 'pcc-and-pce'))"
    {
        error-message
            "The PCEP peer must be PCC";
        description
            "The PCEP peer must be PCC";
    }

    description
        "The last received LSP State
        Database Version Number";
}

container of-list{
    if-feature objective-function;
    must "(/pcep/entity/role = 'pcc')" +
        " or " +
        "(/pcep/entity/role = 'pcc-and-pce'))"
    {
        error-message
            "The PCEP entity must be PCC";
        description
            "The OF-list received on the
```





```
        session";
    }
    uses of-list;

    description
        "Indicate the list of supported OF
        on this session";
}

leaf speaker-entity-id{
    if-feature stateful-sync-opt;
    type string;
    description
        "The Speaker Entity Identifier";
}

leaf discontinuity-time {
    type yang:timestamp;
    description
        "The timestamp value of the time
        when the statistics were last
        reset.";
}

container pcep-stats {
    description
        "The container for all statistics
        at session level.";
    uses pcep-stats{
        description
            "The statistics contained are
            for the current sessions to
            that peer. These are lost
            when the session goes down.
            ";
    }
}

} // pcep-stats
} // peer
} // peers
} // entity
} // pcep-state

/*
 * Notifications
 */
```



```
notification pcep-session-up {
  description
    "This notification is sent when the value of
    '/pcep/pcep-state/peers/peer/sessions/session/state'
    enters the 'session-up' state.";

  uses notification-instance-hdr;

  uses notification-session-hdr;

  leaf state-last-change {
    type yang:timestamp;
    description
      "The timestamp value at the time this session entered
      its current state as denoted by the state leaf.";
  }

  leaf state {
    type pcep-sess-state;
    description
      "The current state of the session.
      The set of possible states excludes the idle state
      since entries do not exist in the idle state.";
  }
} //notification

notification pcep-session-down {
  description
    "This notification is sent when the value of
    '/pcep/pcep-state/peers/peer/sessions/session/state'
    leaves the 'session-up' state.";

  uses notification-instance-hdr;

  leaf session-initiator {
    type pcep-initiator;
    description
      "The initiator of the session.";
  }

  leaf state-last-change {
    type yang:timestamp;
    description
      "The timestamp value at the time this session entered
      its current state as denoted by the state leaf.";
  }

  leaf state {
```



```
        type pcep-sess-state;
        description
            "The current state of the session.
            The set of possible states excludes the idle state
            since entries do not exist in the idle state.";
    }
} //notification

notification pcep-session-local-overload {
    description
        "This notification is sent when the local PCEP entity
        enters overload state for a peer.";

    uses notification-instance-hdr;

    uses notification-session-hdr;

    leaf overloaded {
        type boolean;
        description
            "If the local PCEP entity has informed the peer that
            it is currently overloaded, then this is set to
            true. Otherwise, it is set to false.";
    }

    leaf overload-time {
        type uint32;
        units "seconds";
        description
            "The interval of time that is remaining until the
            local PCEP entity will cease to be overloaded on
            this session.";
    }
} //notification

notification pcep-session-local-overload-clear {
    description
        "This notification is sent when the local PCEP entity
        leaves overload state for a peer.";

    uses notification-instance-hdr;

    leaf overloaded {
        type boolean;
        description
            "If the local PCEP entity has informed the peer
            that it is currently overloaded, then this is set
            to true. Otherwise, it is set to false.";
```



```
    }
  } //notification

notification pcep-session-peer-overload {
  description
    "This notification is sent when a peer enters overload
    state.";

  uses notification-instance-hdr;

  uses notification-session-hdr;

  leaf peer-overloaded {
    type boolean;
    description
      "If the peer has informed the local PCEP entity that
      it is currently overloaded, then this is set to true.
      Otherwise, it is set to false.";
  }

  leaf peer-overload-time {
    type uint32;
    units "seconds";
    description
      "The interval of time that is remaining until the
      peer will cease to be overloaded. If it is not known
      how long the peer will stay in overloaded state, this
      leaf is set to zero.";
  }
} //notification

notification pcep-session-peer-overload-clear {
  description
    "This notification is sent when a peer leaves overload
    state.";

  uses notification-instance-hdr;

  leaf peer-overloaded {
    type boolean;
    description
      "If the peer has informed the local PCEP entity that
      it is currently overloaded, then this is set to true.
      Otherwise, it is set to false.";
  }
} //notification
} //module
```





<CODE ENDS>

## **9. Security Considerations**

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

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

TBD: List specific Subtrees and data nodes and their sensitivity/vulnerability.

## **10. Manageability Considerations**

### **10.1. Control of Function and Policy**

### **10.2. Information and Data Models**

### **10.3. Liveness Detection and Monitoring**

### **10.4. Verify Correct Operations**

### **10.5. Requirements On Other Protocols**

### **10.6. Impact On Network Operations**

## **11. IANA Considerations**

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

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

Registrant Contact: The PCE WG of the IETF.

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



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

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

## **12. Acknowledgements**

The initial document is based on the PCEP MIB [[RFC7420](#)]. Further this document structure is based on Routing Yang Module [[I-D.ietf-netmod-routing-cfg](#)]. We would like to thank the authors of aforementioned documents.

## **13. References**

### **13.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<http://www.rfc-editor.org/info/rfc3688>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", [RFC 5440](#), DOI 10.17487/RFC5440, March 2009, <<http://www.rfc-editor.org/info/rfc5440>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#), DOI 10.17487/RFC6020, October 2010, <<http://www.rfc-editor.org/info/rfc6020>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", [RFC 6991](#), DOI 10.17487/RFC6991, July 2013, <<http://www.rfc-editor.org/info/rfc6991>>.
- [I-D.ietf-pce-stateful-pce] Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", [draft-ietf-pce-stateful-pce-16](#) (work in progress), September 2016.



[I-D.ietf-pce-pce-initiated-lsp]

Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", [draft-ietf-pce-pce-initiated-lsp-07](#) (work in progress), July 2016.

[I-D.ietf-pce-lsp-setup-type]

Sivabalan, S., Medved, J., Minei, I., Crabbe, E., Varga, R., Tantsura, J., and J. Hardwick, "Conveying path setup type in PCEP messages", [draft-ietf-pce-lsp-setup-type-03](#) (work in progress), June 2015.

[I-D.ietf-pce-segment-routing]

Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., Raszuk, R., Lopez, V., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", [draft-ietf-pce-segment-routing-08](#) (work in progress), October 2016.

[I-D.ietf-teas-yang-te]

Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Bryskin, I., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", [draft-ietf-teas-yang-te-04](#) (work in progress), July 2016.

[I-D.ietf-rtgwg-yang-key-chain]

Lindem, A., Qu, Y., Yeung, D., Chen, I., Zhang, Z., and Y. Yang, "Routing Key Chain YANG Data Model", [draft-ietf-rtgwg-yang-key-chain-09](#) (work in progress), September 2016.

## **13.2. Informative References**

- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", [RFC 4655](#), DOI 10.17487/RFC4655, August 2006, <<http://www.rfc-editor.org/info/rfc4655>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", [RFC 6241](#), DOI 10.17487/RFC6241, June 2011, <<http://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", [RFC 6242](#), DOI 10.17487/RFC6242, June 2011, <<http://www.rfc-editor.org/info/rfc6242>>.



- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", [RFC 6536](#), DOI 10.17487/RFC6536, March 2012, <<http://www.rfc-editor.org/info/rfc6536>>.
- [RFC7420] Koushik, A., Stephan, E., Zhao, Q., King, D., and J. Hardwick, "Path Computation Element Communication Protocol (PCEP) Management Information Base (MIB) Module", [RFC 7420](#), DOI 10.17487/RFC7420, December 2014, <<http://www.rfc-editor.org/info/rfc7420>>.
- [I-D.ietf-netmod-routing-cfg] Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", [draft-ietf-netmod-routing-cfg-24](#) (work in progress), October 2016.
- [I-D.ietf-netmod-rfc6087bis] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", [draft-ietf-netmod-rfc6087bis-09](#) (work in progress), October 2016.





**Appendix A. Contributor Addresses**

Rohit Pobbathi  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560066  
India

EMail: rohit.pobbathi@huawei.com

Vinod KumarS  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560066  
India

EMail: vinods.kumar@huawei.com

Zafar Ali  
Cisco Systems  
Canada

EMail: zali@cisco.com

Xufeng Liu  
Ericsson  
1595 Spring Hill Road, Suite 500  
Vienna, VA 22182  
USA

EMail: xufeng.liu@ericsson.com

Young Lee  
Huawei Technologies  
5340 Legacy Drive, Building 3  
Plano, TX 75023, USA

Phone: (469) 277-5838  
EMail: leeyoung@huawei.com

Udayasree Palle  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560066  
India

EMail: udayasree.palle@huawei.com



Xian Zhang  
Huawei Technologies  
Bantian, Longgang District  
Shenzhen 518129  
P.R.China

EMail: zhang.xian@huawei.com

Avantika  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560066  
India

EMail: avantika.sushilkumar@huawei.com

#### Authors' Addresses

Dhruv Dhody (editor)  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560066  
India

EMail: dhruv.ietf@gmail.com

Jonathan Hardwick  
Metaswitch  
100 Church Street  
Enfield EN2 6BQ  
UK

EMail: jonathan.hardwick@metaswitch.com

Vishnu Pavan Beeram  
Juniper Networks  
USA

EMail: vbeeram@juniper.net

Jeff Tantsura  
USA

EMail: jefftant@gmail.com

